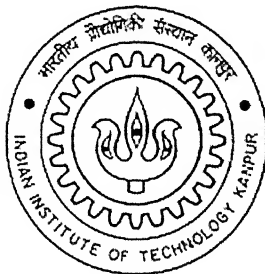


Analysis and Annotation of Cricket Videos

A Thesis submitted in partial fulfillment
of the requirements for the award of Degree of
Master of Technology

by

O. KIRAN KUMAR




DEPARTMENT OF ELECTRICAL ENGINEERING
Indian Institute of Technology Kanpur

August, 2005

CERTIFICATE

It is certified that the work contained in the thesis entitled "*Analysis and Annotation of Cricket Videos*", by O. Kiran Kumar, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

August, 2005



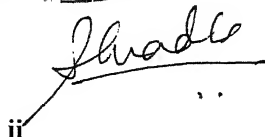
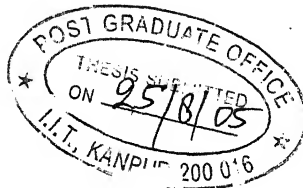
(K. S. Venkatesh)

Asst. Professor,

Department of Electrical Engineering,

Indian Institute of Technology Kanpur

Kanpur – 208 016.



ii

TH
EE/2005/M
K96a

2 SEP 2005/EE

गुरुषोत्तम जी लीनाथ केलकर पुस्तकालय

भारतीय प्रौद्योगिकी संस्थान कानपुर

बुकाचि क्र० A...152769



ABSTRACT

Video can be treated as a sequence of frames, shots, event sequences or stories at different levels of abstraction. Video Shot Detection and Classification is a fundamental step for efficient accessing, retrieval, browsing, highlight generation etc. Summarization of a large amount of video data is one of the popular fields of video research in very recent times. Sports videos in particular have a clear domain knowledge that depends on the particular sport, which helps mainly in shot classification. The aim of the present work is to analyze the cricket videos from the annotation point of view, and to use this analysis in shot classification to automatically classify shots into various semantic categories, and hence generate annotation cues for them. The features used in shot segmentation step are color histograms. We use in shot classification, both color histograms and mean shot energy as features. In addition to automatic shot classification, the work also deals with one of the popular applications of sports video annotation, highlights generation, by detecting the slow motion replay shots.

Dedicated
To
My Grand Parents
Late Smt & Sri O. Hanumantha Rao

ACKNOWLEDGEMENTS

My dream of pursuing higher education at this prestigious and world renowned institution turned into a reality with a great and memorable experience for all through my lifetime, thanks to the people, the environment, the faculty and all the staff of IIT, friends and above all my family members.

I take this opportunity to heart fully thank my thesis supervisor, Dr. K.S. Venkatesh, for accepting me to work under him, and for his invaluable help, guidance and moral support throughout my one year of research under him. It is of no exaggeration to accept the fact that without his kind support and guidance this work would not have been completed.

I express my profound gratitude to the faculty of Dept of Electrical Engg., for the invaluable knowledge they have imparted to me and teaching principles in a most exciting and enjoyable way.

My thanks are due to my parent organization Doordarshan and the kind authorities who have sponsored me for the M.Tech programme.

I also express my thanks to friends Siva balan, A.T. Rajesh and Mahesh for their help in my thesis work, friends Swapnil, Amit, Mayank, Pachori, and colleagues from Doordarshan Manoj, Partha, S.K. Pradhan, Gopal Kumar and S.Rout, for being always cooperative and helpful during my stay here. I thank all my classmates of M.Tech 2003 batch for making my stay at IIT, a comfortable and enjoyable one.

Last but not least, I never forget the endless help and courage provided in all my efforts, by my parents and family members, until now.

CONTENTS

	Page No
List of Figures	ix
List of Tables	x
CHAPTER 1	
INTRODUCTION	
1.1 Need for Automatic Annotation of Cricket Videos	1
1.2 Existing work	2
1.3 Objective of the Thesis	2
1.4 Organization of the Thesis	3
CHAPTER 2	
VIDEO DATABASE MANAGE SYSTEMS (VDBMS)	
2.1 Introduction	4
2.2 Video Databases	6
2.3 Main Components of a VDBMS	7
2.4 Organization of Video Content	8
2.4.1 Video Modeling and Representation	9
2.4.2 Video Segmentation and Summarization	11
2.5 Video Indexing	12
CHAPTER 3	
TEMPORAL SEGMENTATION OF VIDEO DATA	
3.1 Introduction	13
3.2 Editing Effects in Cricket Videos	14
3.3 Related Work	17

3.3.1 Techniques Based on Pixels Difference Metrics(PDMs)	17
3.3.2 Techniques Based on Histogram Difference Metrics (HDMs)	18
3.3.3 Other Techniques	22
3.4 Shot Segmentation Algorithm	22
3.4.1 Introduction	22
3.4.2 Algorithm Outline	23
3.4.3 CUT Detection Algorithm	31
3.4.4 WIPE Detection Algorithm	31
3.4.5 Local Window of Poor Correlations (LWPC) Detection Algorithm	33
3.4.6 DISSOLVE Detection Algorithm	34
3.5 Performance Criteria	36
3.6 Experimental Results	37
3.6.1 Sample Output Shot Data File	38
3.6.2 Shot Detection Results	39
3.7 Performance Evaluation and Computational Efficiency	41
3.7.1 Performance Evaluation	41
3.7.2 Computational Efficiency	41
CHAPTER 4	
ANALYSIS OF CRICKET VIDEOS	43
4.1 Introduction	43
4.2 Typical Positions of Cameras	43
4.3 Semantic Classification of Shots	46
4.4 Complexity of Domain Knowledge	49
CHAPTER 5	
AUTOMATIC ANNOTATION OF CRICKET VIDEOS	53
5.1 Introduction	53
5.2 Existing Work	54
5.3 Automatic Shot Classification	55
5.3.1 Shot Representation	55
5.3.2 Extraction of Shot Templates	55

5.3.3 Shot Classification Algorithm	59
5.3.4 Shot Classification Results	61
5.3.5 Results: Comments	63
5.4 Detection of Slow Motion Replays	63
5.5 Annotation Cues	66
5.6 Applications of Automatic Annotation	67
CHAPTE 6	
CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK	68
6.1 Conclusions	68
6.2 Suggestions for Future Work	69
 BIBLIOGRAPHY	 70

LIST OF FIGURES

	Page No
Fig 2.1	Hierarchy of Database Management Systems 4
Fig 2.2	Visual Information Retrieval blends together many research disciplines 6
Fig 2.3	Block Diagram of a VDBMS 8
Fig 3.1	General Structure of a Video Sequence 14
Fig 3.2	Commonly used editing effects in cricket videos 16
Fig 3.3	Plots of average five frame correlation vs. frame number for a Cut, Dissolve and a Wipe 24
Fig 3.4	Flowchart of Shot Detection Algorithm 26
Fig 3.5	Flowchart for Cut Detection 27
Fig 3.6	Flowchart of Local Window of Poor Correlations (LWPC) Detection Algorithm 28
Fig 3.7	Flowchart for Wipe Detection 29
Fig 3.8	Flowchart for Dissolve Detection 30
Fig 3.9	Shot Detection Results (<i>Men's Cricket Test Match</i>) 40
Fig 4.1	Typical Camera Positions in a Cricket Telecast 44
Fig 4.2	Typical Shots in Cricket Videos 47
Fig 4.3	Sub-classification of shots 48
Fig 4.4	Content Variation within a Shot 50
Fig 4.5	Content Variation across shots having same semantic meaning 51
Fig 4.6	Typical distribution of various shots within a cricket video clip 52
Fig 5.1	Plots of mean shot energy vs. shot number for different categories of shots 57
Fig 5.2	Flowchart of Shot Classification Algorithm 58
Fig 5.3	Performance Comparison of Shot Classification Algorithms 62
Fig 5.4	Shot Classification Results for Slow Motion Replay shots 65

LIST OF TABLES

	Page No.
Table 3.1 Shot detection results for Women cricket match	39
Table 3.2 Shot detection results for Men's cricket match I	39
Table 3.3 Shot detection results for Men's cricket match II	39
Table 4.1 Various shots vs. camera groups used to televise them	45
Table 5.1 Shot Classification Results for various methods	61

CHAPTER 1

INTRODUCTION

1.1 Need for Automatic Annotation of Cricket Videos

Annotation is the process of adding notes or critical comments. Annotation of Sports Videos is one of the popular topics of research in Multimedia Database Management Systems. Since manual annotation of video is subjective, tedious, time consuming and more importantly a costly process there arise the need for automatically annotating the sports videos at different levels of abstraction. This topic is one of the applications of Content Based Video Retrieval (CBVR) systems. Since cricket is becoming more popular in many countries around the world, and as there are large amounts of cricket video archives present with broadcasters, the efficient management of these cricket video archives is a very essential requirement today. Also due to the increasing use of Internet, annotation of sports videos in general has emerged as major application area as users can browse and select the video clips of selected players, sporting actions etc. This application is called as *Video Skimming*. This is also helpful to train sports persons by retrieving the selected portion of the sport in which the sports person is interested.

In addition, the annotation of cricket videos manifests itself in the application of Highlights generation, organizing the selected sporting actions according to player type and best performances of players, type of match, umpire, etc. Also the annotation of cricket videos facilitates the implementation of the so called Virtual Commentator idea, and even special applications in which decisions regarding live sporting actions like whether a batsman is out or not, can be judged when even the third umpire fails to take a decision. The present research work is selected, since very less research work has been done in annotation of cricket videos-worldwide, and due to numerous applications and commercial value that cricket videos have in the countries in which it is played (India, Sri Lanka, Pakistan, West Indies, Australia, England, South Africa, New Zealand, and these days USA and some European Countries like Holland etc).

1.2 Existing work

During the last decade, many researchers around the world have worked on automatic annotation of Soccer [1] [21] and Tennis videos, considering the popularity of these sports in many nations around the world. But very less research work is carried in cricket videos. As far as cricket videos are concerned, in [2] authors tried to automatically detect and classify cricket video shots, using dynamic programming based HMM model. The authors have classified input video shots into four types of cricket events 'Ball Movement', 'Pitch Action', 'Fielding' and 'Wicket Keeping'. This information can be used for Highlight generation, Video summarization etc.

1.3 Objective of the Thesis

Complete annotation of cricket videos is a very exhaustive and time consuming and as well involves with many open research problems like content based image retrieval, face recognition, automatic segmentation of scene into objects etc. However our approach in this thesis is not only to use existing techniques and best algorithms but also try to improve annotation performance wherever possible by exploiting the domain knowledge of cricket videos. Our work proceeds with a view to implement the annotation task at some levels of abstraction like identifying the type of shot, sequence of shots from domain knowledge, identifying the important shots etc. This will help to solve at least some of the problems of manual annotation.

Towards this aim, the first and foremost task in any multimedia data base systems is to model video data, segment the video, classify the segments and hence extract the annotation cues. Keeping this in mind, first we design a computationally efficient and robust shot detection algorithm which can detect both abrupt and gradual transitions.

Though this algorithm uses existing histogram metrics, our contribution has been an entirely new idea of implementing it by considering only five frame correlations. Since cricket videos have clear lighting conditions, the disadvantages of poor performance of histogram methods in dark video sequences do not occur here. Once the shot segmentation task is over, the next step is to identify the different types of shots and classify them into groups. Here we used histogram based methods and motion templates

to classify different types of shots. It is important to mention here that like in any image retrieval systems, no single approach will do the entire classification task in video retrieval systems. Depending on the type of shot, features are extracted by different methods and finally some statistical methods like HMM can be used at higher abstraction to completely classify shots. With this objective in mind, we implement the shot segmentation stage and classify some types of shots that are easily detectable by color histogram methods. Additionally we implemented classification of slow motion replay shots using the shot transitions wipes, and the shot energy as metric.

1.4 Organization of the Thesis

Since our thesis work is selected from the broad area of research, called multimedia data base systems, and in particular, the video data base management systems; we introduce in Chapter 2, the basic concepts and terminology of this area with a view to highlight the major thrust areas of research in this field. Now the important task of almost all video database management based systems, is the exercise of shot segmentation; it is introduced in Chapter 3 right from defining the problem, discussion of existing work as well as our own algorithm. Also we present our shot detection results for three video clips, amounting to around 80,000 frames. In Chapter 4, we give the analysis of cricket videos, which is nothing but the domain knowledge of cricket videos from the automatic annotation point of view. In this chapter, the various types of shots in cricket videos, their characteristics, features, and their significance in annotation are discussed. In Chapter 5 we introduced the tasks of Shot Identification and Classification with a view to extract some annotation cues. We present shot classification results of our algorithm using color, motion and both templates. Also we propose an algorithm to detect slow motion replay shots which are cues for highlights generation. Finally in Chapter 6 we discuss our conclusions and the scope for future work.

CHAPTER 2

VIDEO DATABASE MANAGEMENT SYSTEMS

2.1 Introduction

This chapter aims at introducing the broad area of Video Database Management Systems (VDBMS), from which the present thesis work is selected.

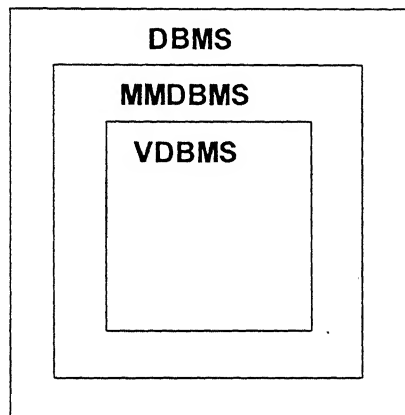


Fig 2.1 Hierarchy of Database Management Systems (DBMS)

The above fig 2.1 indicates that the Video Database Management Systems as a subset of Multimedia Database Management Systems (MMDBMS). The field of distributed multimedia systems has experienced an extraordinary growth during the last decade. Among the many visible aspects of the increasing interest in this area is the creation of huge digital libraries accessible to users worldwide. These large and complex multimedia databases must store all types of multimedia data, e.g., text, images, animations, graphs, drawings, audio, and video clips. Video information plays a central role in such systems, and consequently, the design and implementation of video database systems has become a major topic of interest.

The amount of video information stored in archives worldwide is huge. Conservative estimates state that there are more than 6 million hours of video already stored and this number grows at a rate of about 10 percent a year [3]. Projections estimate that by the end of year 2010, 50 percent of the total digital data stored worldwide will be video and rich media [4]. Significant efforts have been spent in recent years to make the process of video archiving and retrieval faster, safer, more reliable and accessible to users anywhere in the world. Progress in video digitization and compression, together with advances in storage media; have made the task of storing and retrieving raw video data much easier. Evolution of computer networks and the growth and popularity of the Internet have made it possible to access these data from remote locations.

However, raw video data by itself has limited usefulness, since it takes far too long to search for the desired piece of information within a videotape repository or a digital video archive. Attempts to improve the efficiency of the search process by adding extra data called *metadata* to the video contents do little more than transferring the burden of performing inefficient, tedious, and time consuming tasks to the cataloguing stage. The challenging goal is to devise better ways to automatically store, catalog, and retrieve video information with greater understanding of its contents. Researchers from various disciplines have acknowledged this challenge and provided a vast number of algorithms, systems, and papers on this topic during recent years. In addition to these localized efforts, standardization groups have been working on new standards such as MPEG-7, which provide a framework for multimedia content description.

The field of Video Data Bases and their management blends together many areas of research from image processing to computer vision and human activity analysis. Fig 2.2 illustrates the abundant potential of research areas in VDBMS.

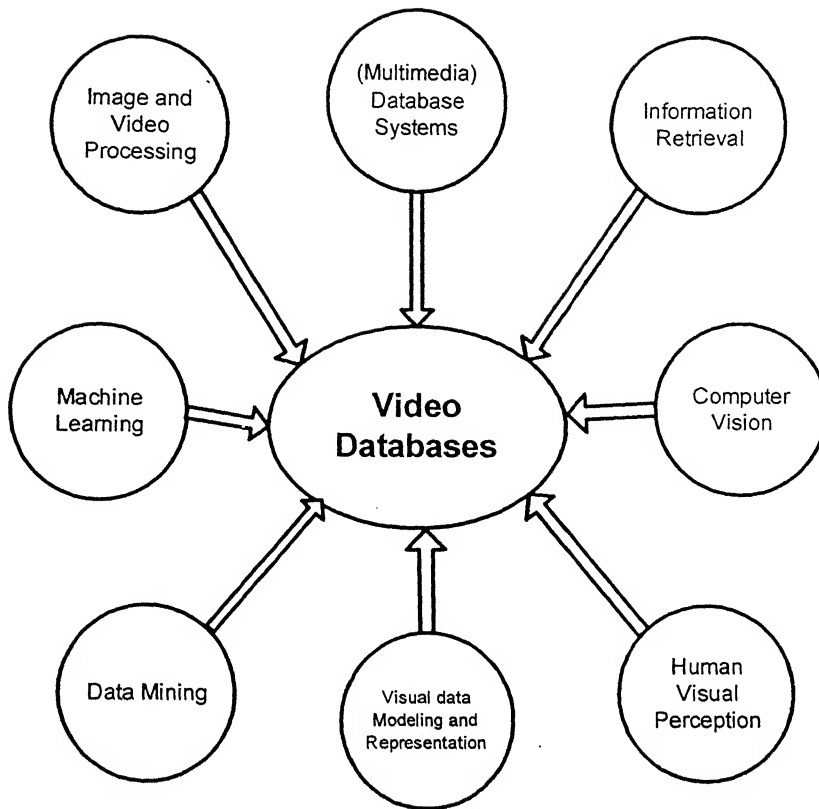


Fig. 2.2 Visual Information Retrieval blends together many research disciplines.

2.2 Video Databases

The challenges faced by researchers when implementing image databases increase even further when one moves from images to image sequences, or video clips, mostly because of the following factors:

- Increased size and complexity of the raw media (video, audio, and text).
- Wide variety of video programs, each with its own rules and formats.
- Video understanding is very much context-dependent.
- The need to accommodate different users, with varying needs, running different applications on heterogeneous platforms.

The primary goal of a Video Database Management System (VDBMS) is to provide pseudo-random access to sequential video data. This goal is normally achieved by dividing a video clip into segments, indexing these segments, and representing the indexes in a way that allows easy browsing and retrieval. Therefore, it can be said that a VDBMS is basically a database of indexes to a video recording [5]. Raw video data alone has limited usefulness and requires some type of annotation before it is catalogued for later retrieval. Manual annotation of video contents is a tedious, time consuming, subjective, inaccurate, incomplete, and perhaps more importantly a costly process. Hence, now, Automatic Annotation of Video contents is a requirement of the researchers all over the world.

Sports Videos in particular have a clear domain knowledge that depends on the type of the sport. Many researchers [1] have worked mostly on Analysis and Annotation of Soccer Videos and Tennis Videos. But Cricket is a very popular sport in Asian subcontinent, Australia, UK etc and these days it is being played in many other European countries and USA as well . It has the highest earnings as compared to any other sport for Television Broadcasters in India. The present work concentrates on automatic analysis of cricket videos as well as extracting some annotation cues which is helpful for a wide variety of applications like Highlight Detection, Efficient Retrieval of Sporting Actions, Efficient management of Sports Video databases etc.. Hence to get more insight into our work, it is relevant here to know something more about the video databases.

2.3 Main Components of a VDBMS:

The main components of VDBMS are shown in fig 2.3. They are explained below:

- *Digitization and Compression:* hardware and software necessary to convert the video information into digital compressed format.
- *Cataloguing:* process of extracting meaningful story units from the raw video data and building the corresponding indexes.
- *Query/Search engine:* responsible for searching the database according to the parameters provided by the user.

- *Digital Video Archive*: repository of digitized, compressed video data.
- *Visual Summaries*: representation of video contents in a concise, typically hierarchical way.
- *Indexes*: pointers to video segments or story units.
- *User interface*: friendly, visually rich interface that allows the user to interactively query the database, browse the results, and view the selected video clips.

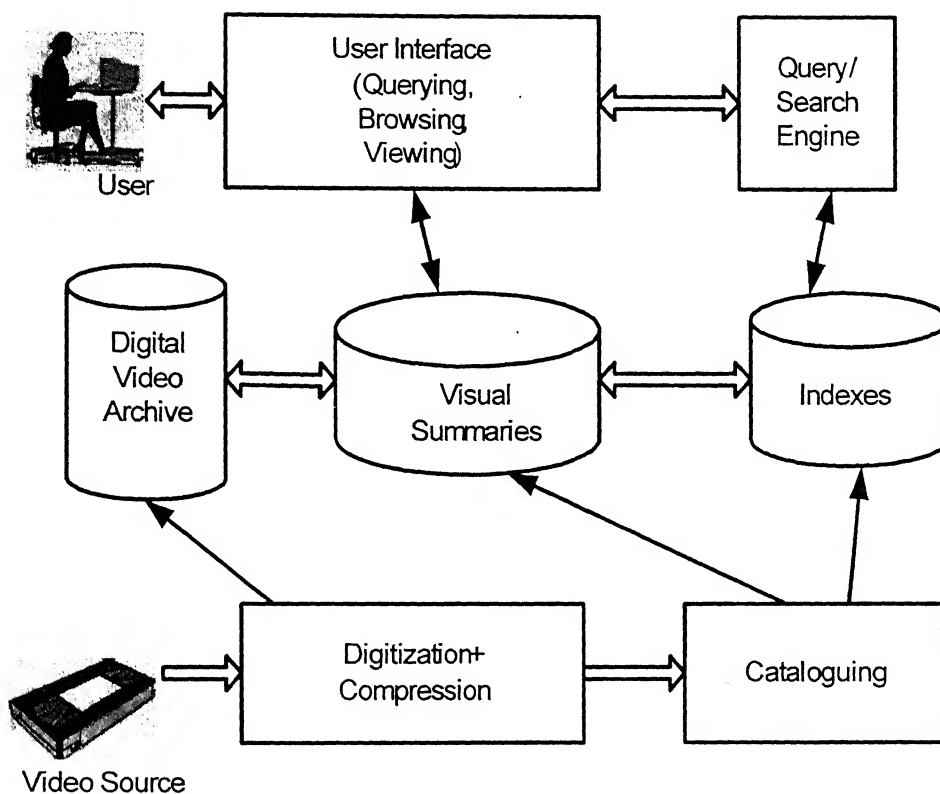


Fig 2.3 Block Diagram of a VDBMS.

2.4. Organization of Video Content

Because video is a structured medium in which actions and events in time and space convey stories, a video program must not be viewed as a non-structured sequence of frames, but instead it must be seen as a document. The process of converting raw video

into structured units, which can be used to build a visual table of contents (ToC) of a video program, is also referred to as Video Abstraction.

Video Abstraction can be divided into two parts:

- Video modeling and representation.
- Video segmentation (parsing) and summarization.

2.4.2 Video modeling and representation

Video modeling can be defined as the process of designing the representation of the video data based on its characteristics, the information content, and the applications it is intended for. This plays a key role in the design of VDBMS, because all other functions are more or less dependent on it.

Video modeling is a challenging task due to large volumes and high dimensionality of video data objects. When referring to contents of video data, distinctions like semantic content, audiovisual content and textual content should be clearly made [6].

Most of the video modeling techniques discussed in the literature adopt a hierarchical video stream abstraction, with the following levels, in decreasing degree of granularity:

- *Key-frame*: most representative frame of a shot.
- *Shot*: sequence of frames recorded contiguously and representing a continuous action in time or space.
- *Group*: intermediate entity between the physical shots and semantic scenes that serves as a bridge between the two.
- *Scene or Sequence*: collection of semantically related and temporally adjacent shots, depicting and conveying a high-level concept or story.
- *Video program*: the complete video clip.

We have modeled our cricket videos using the above model. The concept of *Key frames* arises in the video indexing and retrieval stage, whereby we represent a particular video shot by a frame that represents in a best possible way the particular shot. There are many ways of selecting a key frame. It can be as simple as randomly selecting any frame in the

shot to as complex as selecting the frame that best represents the shot in terms of color histograms, object shape, contours, edge maps etc.

In a cricket video, there are various types of shots, like Batsman Close, Audience, Umpire, Bowler Close etc which will be discussed in chapter 4. These shots in turn have a sequence that will convey some meaning about a higher level semantics, regarding the sporting actions. For example, presence of Bowling, Batting, Fielding shots in a sequence will give the information that a ball is bowled. This is similar to story telling in movies.

A video model should also support annotation of the video program, in other words, the addition of metadata to a video clip. There are broadly three categories of metadata. They are

1. Content-based metadata: e.g., facial features of a sportsperson.
2. Content-descriptive metadata: e.g., the impression of anger or happiness based on facial expression.
3. Content-independent metadata: e.g., name of the sportsperson.

But it is important to note here that, these problems are still open research problems in Image processing. In the present work, the metadata used is at the higher semantic level that a cricket viewer would name a particular sporting action.

Video data models can usually be classified into the following categories [6]:

- *Models based on Video Segmentation*: adopt a two-step approach, first segmenting the video stream into a set of temporally ordered basic units (shots), and then building domain-dependent models upon the basic units.
- *Models based on Annotation Layering*: segment contextual information of the video and approximate the movie editor's perspective on a movie, based on the assumption that if the annotation is performed at the finest grain (by a data camera), any coarser grain of information may be reconstructed easily.
- *Video object models*: extend object-oriented data models to video.
- *Algebraic video data models*: define a video stream by recursively applying a set of algebraic operations on the raw video segment.

- *Statistical models:* exploit knowledge of video structure as a means to enable the principled design of computational models for video semantics, and use machine learning techniques (e.g., Bayesian inference) to learn the semantics from collections of training examples, without having to rely on lower level attributes such as texture, color, or optical flow

2.4.2 Video Segmentation and Summarization

Video segmentation (also referred to as video parsing) is the process of partitioning video sequences into smaller units. Video parsing techniques extract structural information from the video program by detecting temporal boundaries and identifying meaningful segments, usually called shots. The shot (“a continuous action on screen resulting from what appears to be a single run of the cameras”) is usually the smallest object of interest. Shots are detected automatically and typically represented by key-frames. Video segmentation can occur either at a shot level or at a scene level. The former is called as Shot Segmentation, and we will be discussing the topic of shot segmentation and its implementation to cricket videos in detail in chapter 3. The second one called scene-based video segmentation, an alternative to shot detection, consists of automatic detection of semantic boundaries (as opposed to physical boundaries) within a video program. It is a much more challenging task, whose solution requires a higher level of content analysis, and is the subject of ongoing research.

Video Summarization is the process by which a pictorial summary of an underlying video sequence is presented in a more compact form, eliminating or greatly reducing redundancy. Video summarization focuses on finding a smaller set of images (key-frames) to represent the visual content, and presenting these key-frames to the user. Most summarization research involves extracting key-frames and developing a browser-based interface that best represents the original video. An alternative to still-image representation or summarization is the use of video skims, which can be defined as short video clips consisting of a collection of image sequences and the corresponding audio, extracted from the original longer video sequence.

There are two basic types of video skimming:

- *Summary sequences*: used to provide a user with an impression of the video sequence.
- *Highlights*: contain only the most interesting parts of a video sequence.

2.5 Video indexing

Existing work on video indexing can be classified in three categories [6]:

- *Annotation-based indexing*: Annotation is usually a manual process performed by an experienced user, and subject to problems, such as time, cost, specificity, ambiguity, and bias, among several others. A commonly used technique consists of assigning keyword(s) to video segments (shots). Annotation –based indexing techniques are primarily concerned with the selection of keywords, data structures, and interfaces, to facilitate the user’s effort. But keyword based annotation is inherently poor, because keywords do not express spatial and temporal relationships.
- *Feature-based indexing*: They usually rely on image processing techniques to extract key visual features (color, texture, object motion, etc.) from the video data and use these features to build indexes. The main open problem with these techniques is the semantic gap between the extracted features and the human interpretation of the visual scene.
- *Domain-specific indexing*: Techniques that use logical (high-level) video structure models (a priori knowledge) to further process the results of the low-level video feature extraction and analysis stage.

Finally to conclude, in this chapter we have discussed in detail the broad area of research, the Video Data Base Management Systems, in which there are a number of open research problems. Our work is intended mainly to consider the summarization of Cricket videos, so that the large amount of cricket video databases can be managed efficiently. Towards this, we used color information as feature for both shot segmentation and shot classification thus indexing shots and then to aim towards domain specific indexing. In addition to this, we also used shot energy derived from difference images of consecutive frames, as metric to detect slow motion replay shots.

CHAPTER 3

TEMPORAL SEGMENTATION OF VIDEO DATA

3.1 Introduction

Temporal segmentation of video data 'or' Video Shot Segmentation is the process aimed at the detection and classification of transitions between consecutive sequences of frames that are semantically homogeneous and characterized by spatiotemporal continuity. These sequences, generally called camera-shots, constitute the basic units of a video indexing system. In fact, from a functional point of view, temporal segmentation of video data can be considered as the first step of the more general process of content-based automatic video indexing. Although the principal application of temporal segmentation is in the generation of content based video databases, there are other important fields of application. For example in video browsing, automatic summarization of sports video or automatic trailer generation of movies, temporal segmentation is implicitly needed.

It is important to point out the conceptual difference between the operation of an automatic tool aimed at the detection and classification of the information present in a sequence and the way a human observer analyzes the same sequence. While the human observer usually performs a semantic segmentation starting from the highest conceptual level and then going to the particular, the automatic tool of video analysis, in a dual manner, starts from the lowest level, i.e., the video bit stream, and tries to reconstruct the semantic content. For this reason, operations that are very simple and intuitive for a human may require the implementation of quite complex decision making schemes. For example, consider the process of archiving an episode of a TV show. The human observer probably would start annotating the title of the episode and its number, then a general description of the scenes and finally the detailed description of each scene. On the other side the automatic annotation tool, starting from the bit stream, tries to determine the structure of the video based on the homogeneity of consecutive frames and then can extract semantic information.

If we assume a camera-shot as a sequence of semantically homogeneous and spatiotemporally continuous frames then the scene can be considered as a homogeneous sequence of shots and the episode as a homogeneous sequence of scenes. In practice, the first step for the automatic annotation tool is the determination of the structure of the video based on camera-shots, scenes, and episodes as depicted in fig 3.1 below.

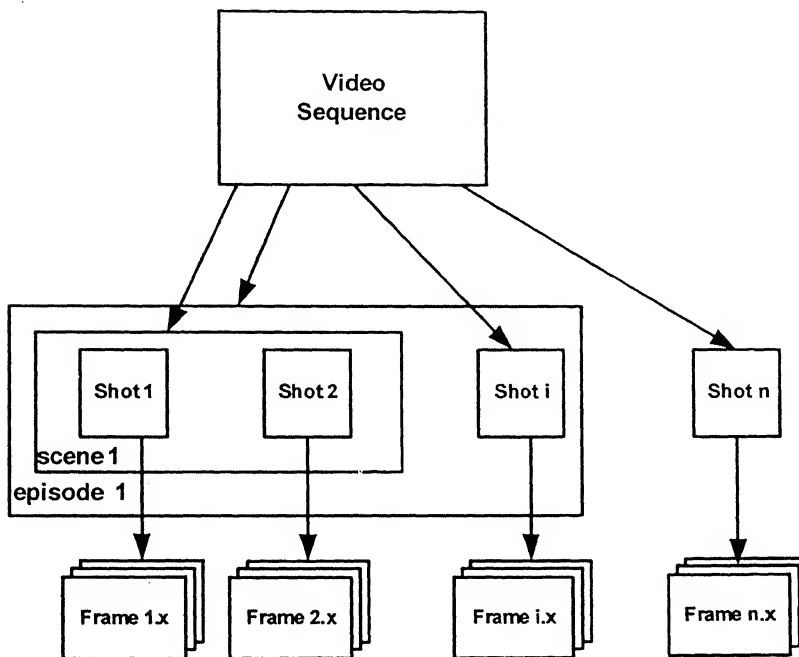


Fig 3.1 General Structure of a Video Sequence

Hence the aim of any Temporal Segmentation technique is :

- To decompose the video into camera shots.
- To identify the type of transition.

3.2 Editing Effects in Cricket Videos

In a video sequence the transitions between camera-shots are called the editing effects and they can be of different typologies. Even though in many cases the detection of a transition is sufficient, in some cases it is important at least to determine if the transition is abrupt or gradual. But in cricket video annotation, it is very helpful if we also know

the type of transition as well, since we can extract some annotation cues from them. For example, slow motion replays are generally bounded by two wipe effects and needless to say these slow motion replays are the cues for extracting Highlight actions, like a boundary, a six, or a wicket etc. The most popular editing effects used in Cricket Videos are Cut, Dissolve, and a Diagonal Wipe. Now we shall discuss each one of them in detail. Fig 3.2 illustrates the various editing effects.

Cuts: Cuts are abrupt transitions which involve only two frames (one for each shot). This type of transition occurs when a sudden change in camera takes place. These are the most frequently used transitions in cricket videos.

Dissolves: Dissolve is a very common editing effect and consists of a gradual transition between two shots where the first one slowly disappears and, at the same time and at the same place, the second one slowly appears. In Dissolve the superposition of two shots is obtained by changing the intensity of the frames belonging to the two shots. The no of frames typically involved in dissolve is between 4 to 7 frames. In cricket videos, it is observed that the dissolves are used mainly when there is idle time between sporting actions, like from scoreboard shot to other shots or between any two fielder close shots/ bowler close shots etc. The other types of dissolves are fade-in and fade-out, which occur at the beginning and end of video respectively. Fade-in and Fade-out consist of the transition between the solid background and shot, and vice-versa respectively.

Wipes: Wipes are the family of gradual transitions where the first shot is progressively covered with the second one following a well defined trajectory. The trajectory can be diagonal/longitudinal/traversal/radial etc. In the case of wipes, the superposition of the two shots is obtained by changing the spatial positions of the boundary between the simultaneously displayed frames of the old and the new shot. The no of frames typically involved in a wipe is 9 to 11. In cricket videos, diagonal wipes are very commonly used. They are used whenever a highlight action occurs and a replay is to be shown. Because of the presence of the wipe effects only, the viewer will be able to identify replay actions easily. *Hence wipes form vital tools in extracting highlights in cricket videos.*

A) CUT



i^{th} Frame



$(i+1)^{th}$ Frame

B) DIAGONAL WIPE



j^{th} Frame

$(j+1)^{th}$ Frame

$(j+3)^{rd}$ Frame



$(j+5)^{th}$ Frame

$(j+7)^{th}$ Frame

$(j+9)^{th}$ Frame

C) DISSOLVE



$(k+1)^{th}$ Frame

$(k+3)^{rd}$ Frame

$(k+5)^{th}$ Frame

$(k+6)^{th}$ Frame

Fig 3.2 Commonly Used Editing Effects in Cricket Videos

3.3 Related Work

In the past decade, researchers have made considerable contributions in stand-alone image analysis and recognition to allow the partitioning of a video source into segments. Basically, temporal segmentation algorithms are based on the evaluation of the quantitative differences between successive frames and on some kind of thresholding. In general an effective segmentation technique must combine an inter frame metric that is computationally simple and able to detect video content changes with robust decision criteria. In subsections 3.3.1-3.3.3, we will review some of the most significant approaches proposed in recent years and discuss their strength and limitations. These techniques also constitute the building blocks of more sophisticated segmentation methodologies and tools.

3.3.1 Techniques Based on Pixels Difference Metrics (PDMs)

Metrics classified as PDMs (pixels difference metrics) are based on the intensity variations of pixels in equal position in consecutive frames. A basic PDM is the sum of the absolute differences of intensity of the pixels of two consecutive frames [9]. In particular, indicating with $Y(x, y, j)$ and $Y(x, y, k)$ the intensity of the pixels at position (x, y) and frames j and k , the metric can be expressed in the following way:

$$\Delta f = \sum_x \sum_y |Y(x, y, j) - Y(x, y, k)|$$

where the summation is taken all over the frame.

The same authors propose other metrics based on first and second order statistical moments of the distributions of the levels of intensity of the pixels. Indicating with μ_k and σ_k respectively, the values of mean and standard deviation of the intensity of the pixels of the frame k , it is possible to define the following inter frame metric between the frames j and k .

$$\lambda = \frac{\left[\frac{\sigma_j + \sigma_k}{2} + \left(\frac{\mu_j - \mu_k}{2} \right)^2 \right]^2}{\sigma_j \sigma_k}$$

This metric has been also used in [8], [14], and is usually named *likelihood ratio*, assuming a uniform second order statistic. Associating a threshold to a metric, it is possible to detect a transition whenever the metric value exceeds the threshold value. As it was pointed out in [9], PDMs offer the best performance for the detection of abrupt transitions. In general all the PDMs are particularly exposed to the effects of noise, camera/object movements or sudden lighting changes, leading to temporally or spatially localized luminance perturbations, and then to a potentially large number of false transitions.

3.3.2 Techniques Based on Histogram Difference Metrics (HDMs)

Since we have used Histograms all through our work in Shot Segmentation and Shot Classification, as well, so we will be reviewing the related work using these techniques in detail:

Metrics classified as HDM (histograms difference metrics) are based on the evaluation of the histograms of one or more channels of the adopted color space. As it is well known that the histogram of a digital image is a measure that supplies information on the general appearance of the image. With reference to an image represented by three color components, each quantized with 8 bit/pixel, a three-dimensional histogram or three one-dimensional histograms can be defined. Although the histogram does not contain any information on the spatial distribution of intensity, the use of inter frame metrics based on image histograms is much diffused because it represents a good compromise between computational complexity and the ability to represent image content. Also, the histogram comparison method is less sensitive to object motion than the pixel-pair wise comparison algorithms, using various histogram difference measures.

In what follows is the review of some of the most popular HDM metrics. In all the equations M and N are respectively the width and height (in pixels) of the image, j and k are the frame indices, L is the number of intensity levels and $H(j, i)$ the value of the histogram for the i^{th} intensity level at frame j .

A commonly used metric [10] is the *bin-to-bin* difference, defined as the sum of the absolute differences between histogram values computed for the two frames:

$$f_{db2b}(j, k) = \frac{1}{2MN} \sum_{i=0}^{L-1} |H(j, i) - H(k, i)|$$

The metric can easily be extended to the case of color images, computing the difference separately for every color component and weighting the results. For example, for a RGB representation we have:

$$f_{drgb}(j, k) = \frac{r}{s} f_{db2b}(j, k)^{(red)} + \frac{g}{s} f_{db2b}(j, k)^{(green)} + \frac{b}{s} f_{db2b}(j, k)^{(blue)}$$

where r , g and b are the average values of the three channels and s is given by :

$$s = \frac{(r + g + b)}{3}$$

Another metric, used for example in [10], [7] is called *intersection* difference and is defined in the following way:

$$f_{dint}(j, k) = 1 - \frac{1}{MN} \sum_{i=0}^{L-1} \min[H(j, i), H(k, i)]$$

In other approaches [13], the *chi-square* test has been used, which is generally accepted as a test useful to detect if two binned distributions are generated from the same source:

Also the correlation between histograms is used:

$$f_{dcorr}(j, k) = 1 - \frac{\text{cov}(j, k)}{\sigma_j \sigma_k}$$

where $\text{cov}(j, k)$ is the covariance between frame histograms and is given by:

$$\text{cov}(j, k) = \frac{1}{L} \sum_{i=0}^{L-1} [H(j, i) - \mu_j][H(k, i) - \mu_k]$$

and μ_j and σ_j represent the mean and the standard deviation, respectively, of the histogram of the frame j given by:

$$\mu_j = \frac{1}{L} \sum_{i=0}^{L-1} H(j, i) \quad \text{and} \quad \sigma_j = \sqrt{\frac{1}{L} \sum_{i=0}^{L-1} [H(j, i) - \mu_j]^2}$$

All the metrics discussed so far are global, i.e., based on the histogram computed over the entire frame. Some authors propose metrics based on histograms computed on sub frames. For example, in [18] a rectangular 6x4 frame partitioning is used. i.e., each frame is subdivided into 6 horizontal blocks and 4 vertical blocks. Then an HDM is used to compute difference between corresponding blocks in consecutive frames. The global metric is finally obtained adding up the contribution of all the sub-regions or blocks.

It is important to mention here that, the number of blocks should not be increased too much for two reasons:-

- This will slow down the cut detection process.
- The algorithm will tend to simulate the pixel pair-wise histogram algorithm.

The possibility of detecting false camera cuts will then increase and the efficiency will decrease in case of motion resulting from quick camera operation or large optical flow.

However, the number of blocks should not be too small as well, in order to avoid the overall distribution problem of missing true visual cuts.

Both PDM and HDM techniques are based on the computation of a similarity measure of two subsequent frames and on comparison of this measure with a threshold. The choice of the threshold is critical, as too low threshold values may lead to false detections and too high threshold values may cause the opposite effect of missed transitions. To limit the problem of threshold selection, several techniques have been proposed. For example, in [18] a short-term analysis of the sequence of inter frame differences is proposed to obtain temporally localized thresholds. The analysis is performed within a temporal window of appropriate size, for example 7 frames. This approach is more robust to local variations of brightness and camera/object motion. In detail naming D_k the difference metric between the frames k and $k-1$ and W_k the short-term temporal window centered on the frame k , we have:

$$\rho(k) = \frac{D_k}{M_k} \quad \text{where} \quad M_k = \max_{k \in W_k} \{D_k\}$$

If $\rho(k)$ exceeds a threshold, computed experimentally, then an abrupt transition at frame k is detected.

The techniques presented above, based on PDMs or HDMs, are mainly suited for the detection of abrupt transitions. But these techniques are not suitable for detecting gradual transitions like wipes and dissolves or fades. For example, in [18], the authors propose a technique to detect fading effects based on a linear model of the luminance L in the CIE L^*u^*v color space. Assuming that chrominance components are approximately constant during the fading, the model for a fade-out is:

$$L(x, y, t) = L(x, y, t_0) \left(1 - \frac{t - t_0}{d} \right); \quad t \in [t_0, t_0 + d]$$

where $L(x, y, t)$ is the luminance of the pixel at position (x, y) and time t , t_0 is the time of beginning of the fading effect and d its duration.

Twin-Comparison Method: In [16] a simple and effective two-threshold technique based on HDM is reported. The two thresholds respectively detect the beginning and end of the transition. The method, called *twin-comparison*, takes into account the cumulative differences between frames of the gradual transition. In the first pass a high threshold T_h is used to detect cuts; in the second pass a lower threshold T_l is employed to detect the potential starting frame F_s of a gradual transition. F_s is then compared to subsequent frames. An accumulated comparison is performed as during a gradual transition this difference value increases. The end frame F_e of the transition is detected when the difference between consecutive frames decreases to less than T_l , while the accumulated comparison has increased to value higher than T_h . If the consecutive difference falls below T_l before the accumulated difference exceeds T_h , then the potential start frame F_s is dropped and the search continues for other gradual transitions.

Finally, techniques based on higher level image features have also been tried. For example in [19] the analysis of intensity edges between consecutive frames is used.

During a cut or a dissolve, new intensity edges appear far from the locations of the old edges. Similarly, old edges disappear far from the location of new edges. Thus, by counting the entering and exiting edge pixels, cuts, fades and dissolves may be detected and classified.

3.3.3 Other Techniques

There are several other techniques for solving the problem of Shot Segmentation.

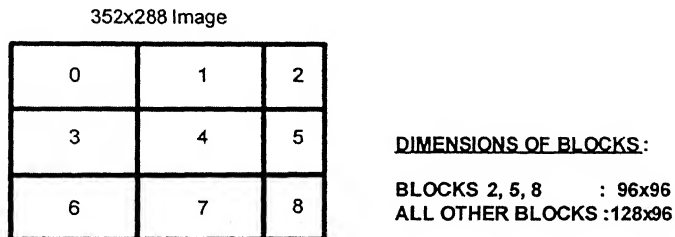
Some of them are:

- Techniques based on MPEG Compressed Video[11][12][15][17]
- Many other researchers have tried to organize one or more of the previously described basic algorithms within more general frameworks, with the aim of defining and implementing more robust techniques.
- Techniques based on use of Multilayer Perceptron (MLP) , Wavelet Based etc.

3.4 Shot Detection Algorithm

3.4.1. Introduction

Here we will be discussing the Shot Detection Algorithm that is proposed by us which is computationally very efficient without compromise on overall system performance and which can detect both abrupt and gradual transitions. Let us first have a look at the salient points of our algorithm:



- Image Correlation is found by evaluating *3D Block Histogram Comparison* for all the blocks of two frames, where the number of blocks per frame is nine. The overall image correlation is found by averaging the nine block correlations. It is

important to note that in this chapter, except in Cut Detection module, image correlation means the image correlation of frames separated by five frames apart.

- Our experiments were carried out using computationally efficient routines from *Intel® Open Source Computer Vision Library*. The histogram correlation is performed by this formula:

Let H_1 and H_2 be histograms of the same ROI of two images to be compared.

Let d be the correlation value lying between 0 and 1.

$$d(H_1, H_2) = \frac{\sum_{I=0}^{N-1} H_1^1(I) H_2^1(I)}{\sqrt{\sum_{I=0}^{N-1} [H_1^1(I)]^2 \sum_{I=0}^{N-1} [H_2^1(I)]^2}}$$

$$\text{Where } H_k^1(I) = H_k(I) - \frac{1}{N} \sum_{J=0}^{N-1} H_k(J)$$

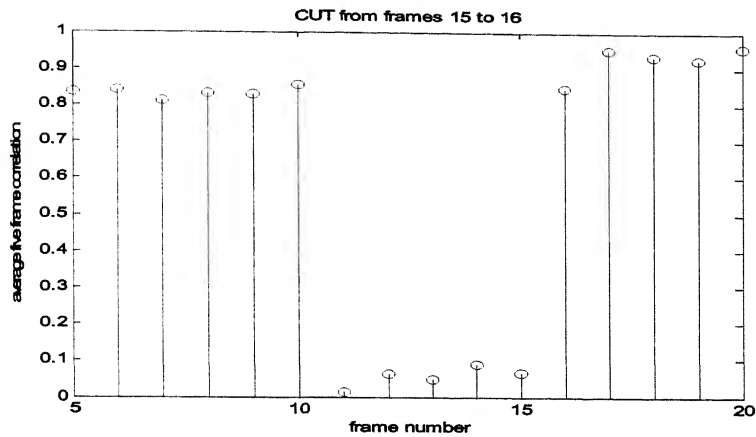
where N represents the number of histogram bins

- No of bins used for each dimension are 20. This implies 8000 bins for each block, and altogether there 72000 bin comparisons involved for evaluating every correlation between any two frames.
- We have used correlation statistics of frames separated by a temporal step, say 5 in detecting various types of shot transitions. We will see later that this is the main step which reduces the computational cost of our shot detection algorithm significantly.

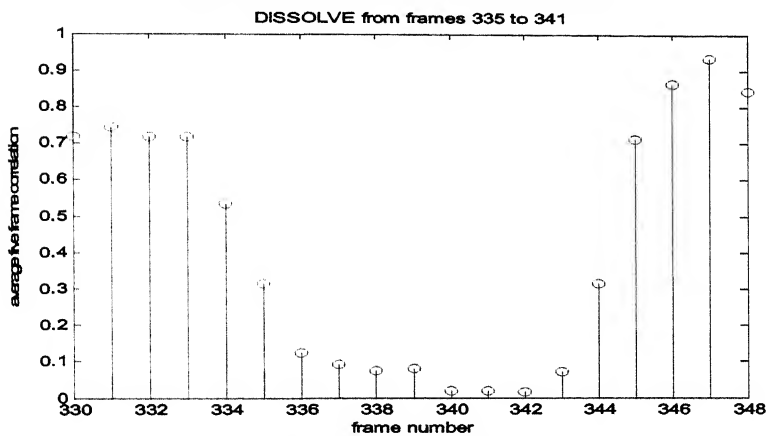
3.4.2 Algorithm Outline

In Fig. 3.3 are plotted the values of average correlations obtained for frames separated by five frames apart as explained in section 3.4.1 above. The plots are shown for three different types of shot transitions which are most commonly used in Cricket Videos. These simple plots give us an idea that when ever there is a shot transition, then the correlation values are well below some threshold, say 0.45, for some window of frames. Hence the correlation statistics are used in fixing various thresholds that we used in our algorithm of shot detection. This type of analysis resulted in our shot detection algorithm to be computationally efficient, which is will be discussed in section 3.7.2.

A) CUT FROM 15 TO 16



B) DISSOLVE FROM 335 TO 341



C) WIPE FROM 225 TO 234

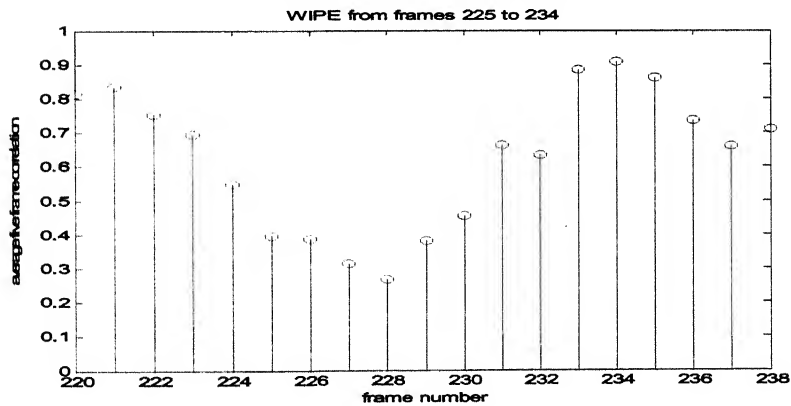


Fig 3.3 Plots of Average five frame correlation vs frame number for
A)Cut B)Dissolve and C) Wipe

Fig 3.4 represents the overview of the proposed shot detection algorithm which will detect Cuts, Dissolves and Diagonal Wipes. The outline of the Shot Segmentation algorithm is given below:

Step 1: Capture the video frame wise, where the frame index is named as f

Step 2: For every frame compute the average correlation with the frame that is at a distance of some temporal step (say 5), from all the block correlations as explained in section 3.4.1. Let us name the average correlation value by $corr5$.

Step 3: If $corr5 \leq T_{main}$, then the algorithm checks for presence of any shot transition.

Else $f \leftarrow f + 5$ (i.e., to capture next fifth frame) and *goto Step 1*

//Now we will check for presence of shot transition and identify the type.//

Step 4: If $corr5 \leq T_{cl}$, then call **CUT Detection Algorithm**. Within this cut detection algorithm, we also check out if any dissolves are detected as cuts.

Step 5: If a cut is found in *Step 4*, then f will now point to starting frame of new shot, and *goto Step 1*. (Even if a dissolve is found which is falsely detected as some consecutive cuts, then also f will point to starting frame of new shot, and *goto Step 1*.)

Else call Local Window of Poor Correlations Detection Algorithm (LWPC) to determine frames $wbegin$ and $wend$ within which $corr5 \leq T_{lw}$ ($T_{lw} \leq T_{main}$)

Step 6: Call **WIPE Detection Algorithm** which will find if a wipe starting from any of the frames between $wbegin$ and $wend$ found from *Step 5*, is present or not.

Step 7: If a Wipe is found in *Step 6*, then f will now point to starting frame of new Shot and *goto Step 1*

Else call **DISSOLVE Detection Algorithm**.

Step 8: If a Dissolve is found from *Step 7* then f will now point to starting frame of new shot and *goto Step 1*

Else $f \leftarrow f + 5$ and *goto Step 1*

Step 9: Repeat all the above steps until all the N frames in the video clip are considered.

Fig 3.4 FLOW CHART OF SHOT DETECTION ALGORITHM

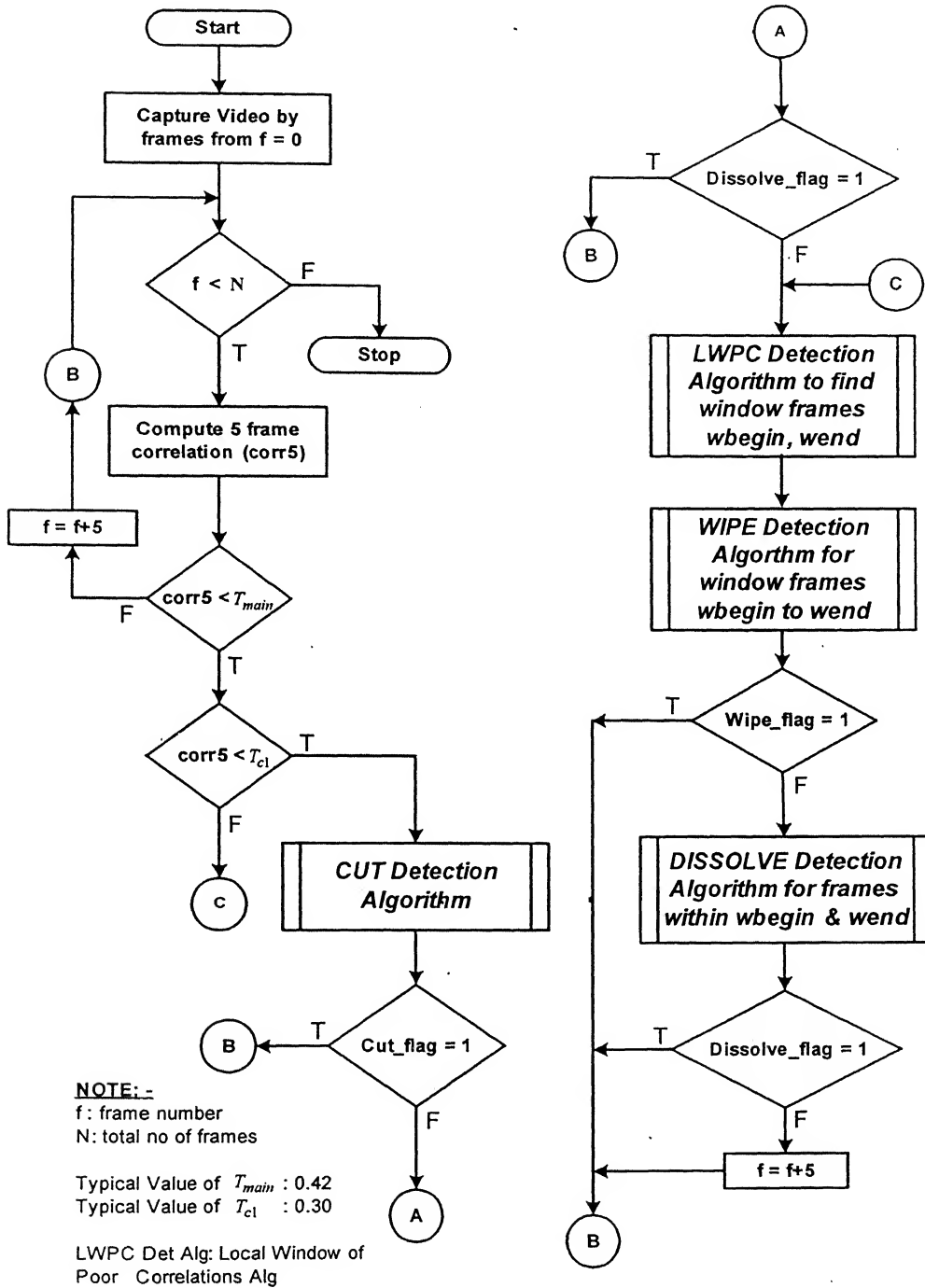


Fig 3.5 FLOWCHART FOR CUT DETECTION

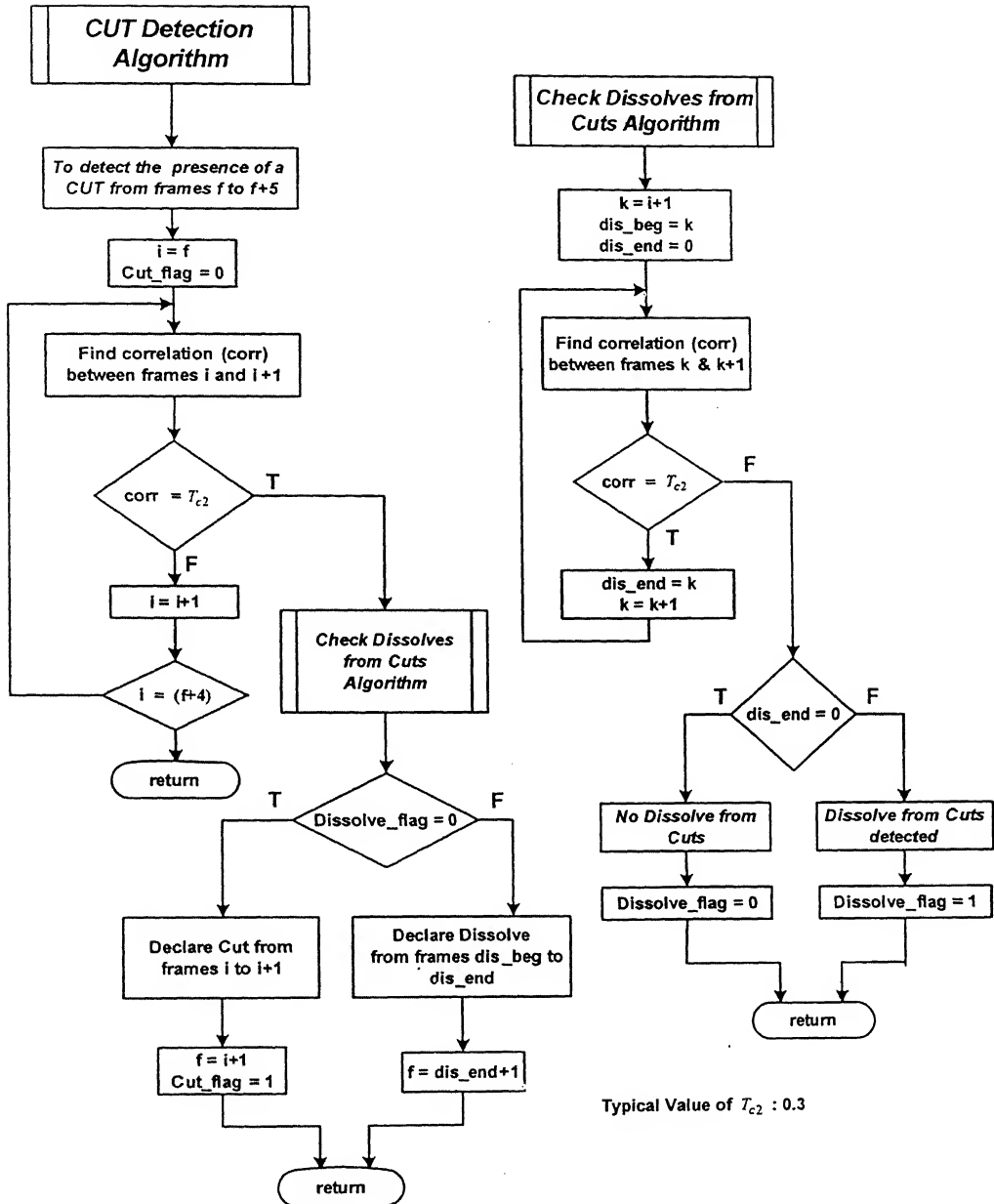


Fig 3.6 FlowChart of Local Window of Poor Correlations (LWPC) Detection Algorithm

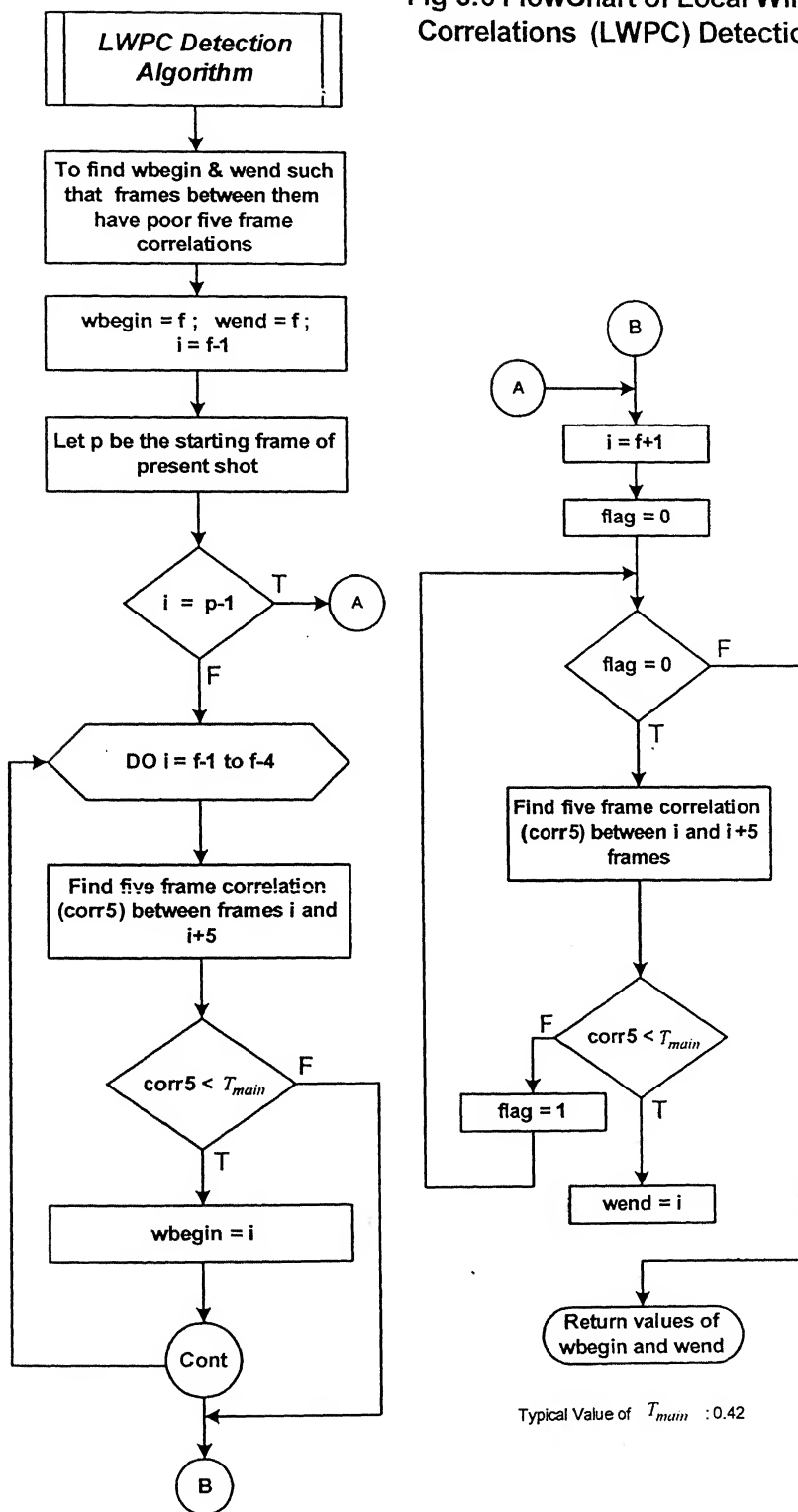


Fig 3.7 FLOWCHART FOR WIPE DETECTION

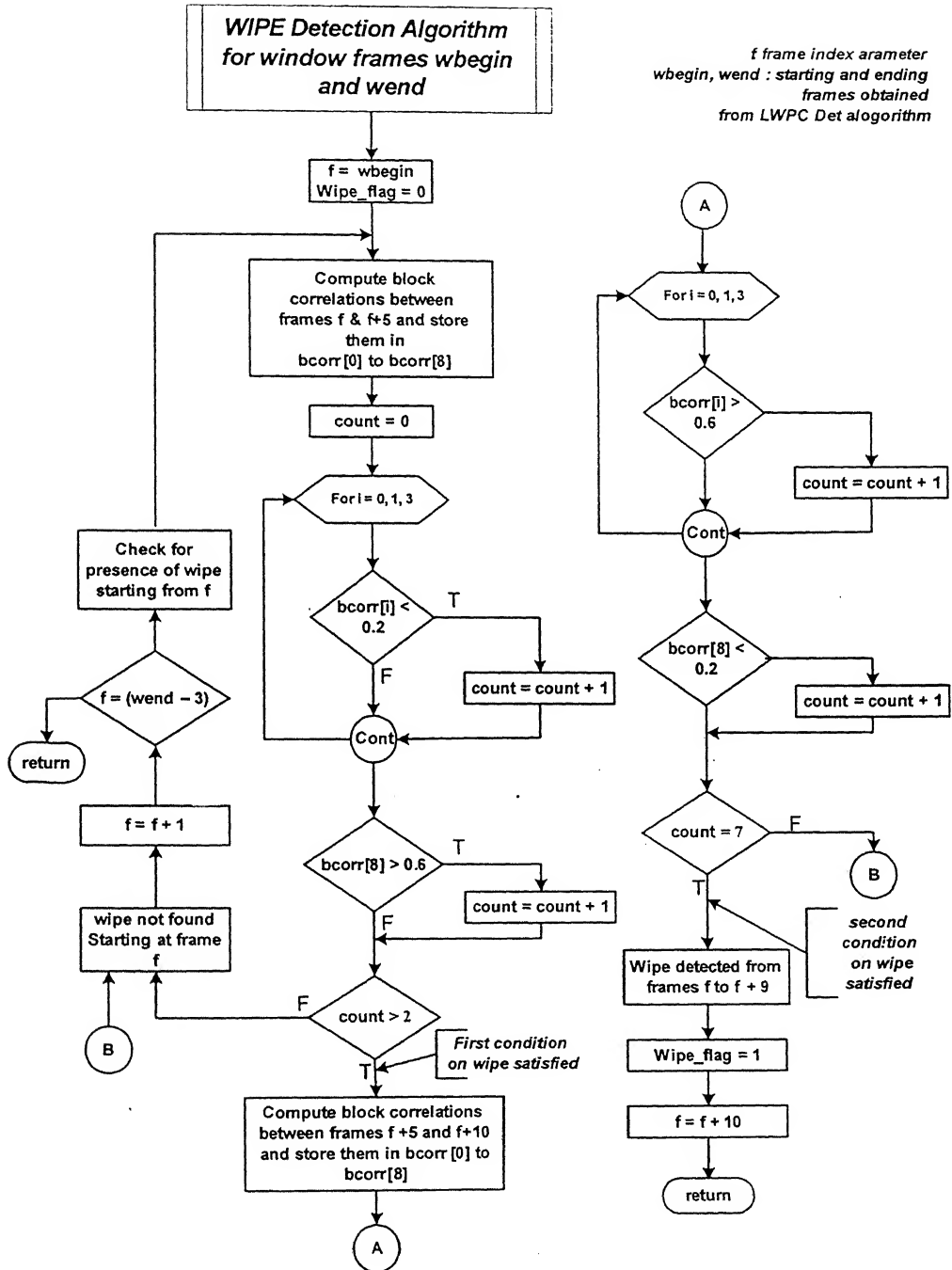
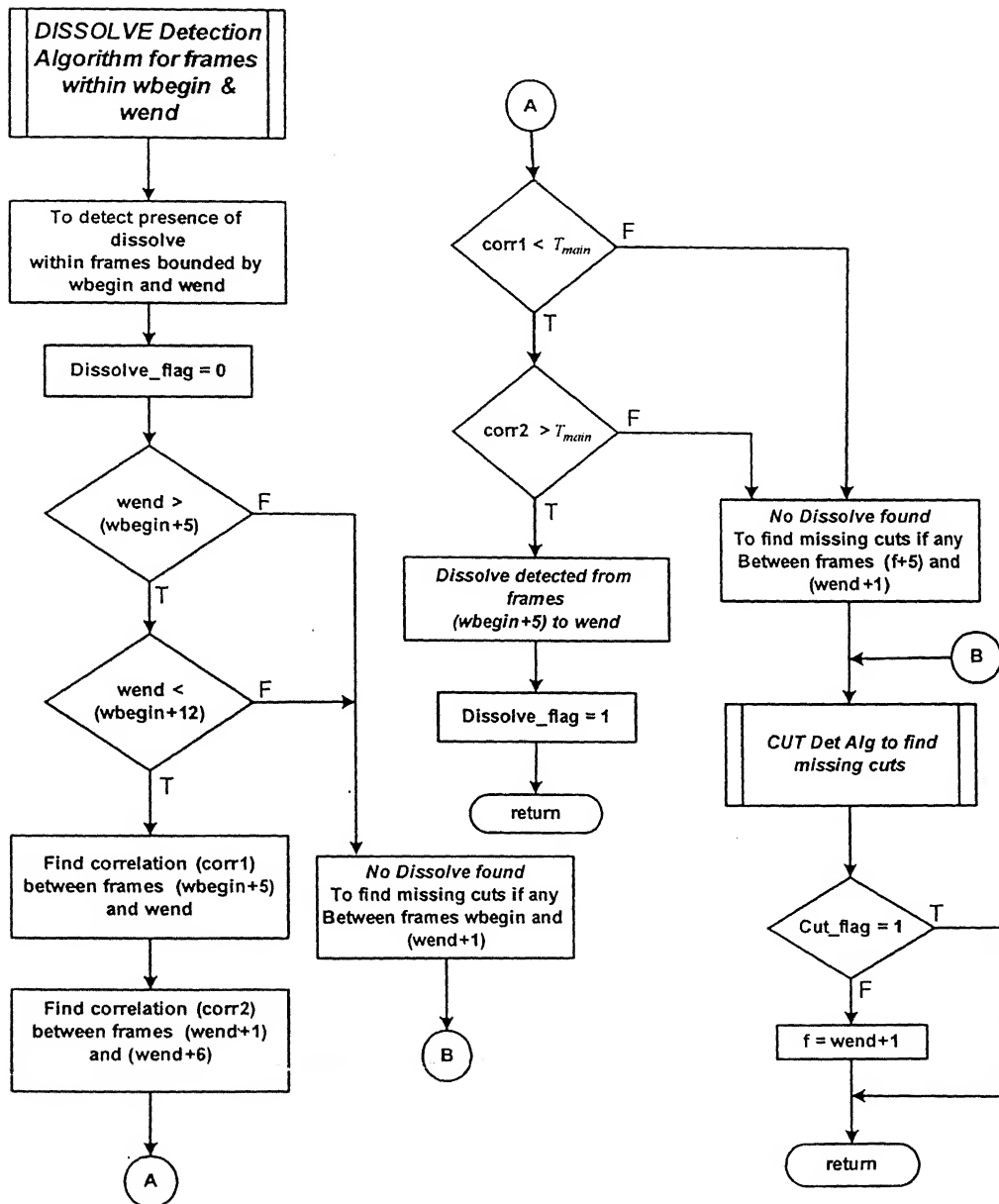


Fig 3.8 FLOWCHART FOR DISSOLVE DETECTION



3.4.3 CUT Detection Algorithm

Fig 3.5 represents the flowchart of Cut Detection Algorithm.

The algorithm for cut detection is a two pass technique. In the first pass we check if there is a possibility of a cut between frames f and $f+5$. For this we check if $corr5 \leq T_{c1}$ from Step 4 in section 3.4.2. Once this condition is satisfied, then we look for two adjacent frames $i, i+1$ in between the frames f and $f+5$, such that their average correlation is low. i.e., below some threshold, T_{c2} , where $T_{c2} \leq T_{c1}$. If the second condition is also satisfied then a cut is said to be present between $i, i+1$, provided there is no dissolve occurring with this cut and subsequent cuts. Because some times due to fast dissolves, the thresholds are such that, dissolve is detected as successive shots of length one and all transitions being detected as cuts. To avoid such false positives we have added additional block to our cut detection algorithm which is called as *Check Dissolves from Cuts Algorithm*. This algorithm is called once two conditions of cut are satisfied. Suppose $i, i+1$ frames make a cut, then $i+1, i+2$ frames are checked for cut. For $i, i+1$ to have a cut transition, the required condition is that $i+1, i+2$ should not have a cut. *Hence we have not only detected correct cuts and avoided false cuts from dissolves but at the same time we have improved no of correct detections for dissolves.* Now, the respective flag for Cut or Dissolve is set accordingly, and f will point to new shot transition.

3.4.4 WIPE Detection Algorithm

Fig 3.7 represents the Wipe Detection Algorithm.

This algorithm is invoked in the main shot detection algorithm whenever both the conditions $corr5 \leq T_{min}$ and $Cut_flag = 0$ are satisfied. This means that the five frame correlation indicates a shot transition, and we have already found that there is no cut. So now we have to check for gradual transitions. The algorithm for Wipe is simple and robust. By wipe, we always mean diagonal wipe, which is used in cricket videos. From the definition of Wipe as given in section 3.2 and from fig 3.2. (b), it is clear that frames at the wipe boundaries differ by around 50 % of the image (diagonal-wise), with middle

frame of wipe. To be more specific, first and fifth frames in a Wipe transition will have poor correlation in blocks 0, 1, and 3 and good correlation in blocks 6, 7 and 8. The fifth and tenth frames in a Wipe transition will have good correlation in blocks 0, 1, and 3 and poor correlation in blocks 6, 7, and 8 i.e., this is converse to earlier condition. It is difficult to say clearly about the correlations in blocks 4 and 5 due to practical difficulties in locating exact start of wipe action, and rate of wiping action. So, here we can have many rules be it hard, or soft which are based on above block correlations to detect a wipe.

We have used the following method to detect wipes. This is also two pass technique. First we find the block correlations of frames f and $f + 5$. and check if blocks 0, 1, and 3 have poor correlation and block 8 has good correlation. We count how many of these four conditions are satisfied. *If the count is greater than 2, then the first condition on Wipe is satisfied.* Now we find block correlations of frames $f + 5$ and $f + 10$, and we check for good correlation in blocks in 0,1 and 3 and poor correlation in block 8. We again count the no of conditions satisfied in second case and accumulate to the earlier count value. A wipe is said to be present between frames f and $f + 10$ if count is at least 7, i.e., at least seven out of eight conditions are satisfied. *This is the second condition on Wipe.* If second condition is satisfied, then the wipe flag is set, and frame index is set to $f + 11$, to point to next shot transition. However if the condition on count fails either in pass one or pass two of algorithm, then there is no wipe, and wipe flag is reset. In such a case, we have to search for presence of dissolve. The thresholds selected for poor correlation is 0.2, and for good correlation is 0.6.

However, the wipe is checked between frames indexed by f varying from $wbegin$ and $wend$ obtained from LWPC Detection Algorithm. The reason for it and the LWPC Detection Algorithm is explained below in section 3.4.2.3.

3.4.5 Local Window of Poor Correlations (LWPC) Detection Algorithm

The purpose of this module is to find the all the frames within a local window whose five frame correlations are less than some threshold, T_{lw} where $T_{lw} \leq T_{main}$. T_{main} is the main threshold (around 0.42 typically) which is used to decide, whether we need to check a shot transition between two frames separated by five frames apart. The LWPC Detection Algorithm is called when there is no cut or dissolve found from Cut Detection Algorithm. *The necessity of this algorithm is due to the following reasons:*

1. **Problem in Wipe Detection:** Though the above algorithm for detection of wipes is hard enough to reduce false positives, but it has one drawback. It needs that we track exactly the first frame of the wipe action. Then only the first and fifth frames of wipe satisfy the condition on $count > 2$ in pass one. Similarly in pass two the condition $count > 7$ gets satisfied. But if our search tracks any other frame of wipe and tries to find a wipe, then our algorithm fails. This is illustrated by an example. Let us consider that there is a wipe present between 250 to 260 frames. Now if our frame index parameter f is 247. If the five frame correlation between frames 247 and 252 is just above T_{main} , then this is an indication that there is not shot transition between 247 and 252, and that the frame index f is modified to 252 to search new shot transition. This time the correlation between 252 and 257 will be below T_{main} , because there is a wipe between 250 to 260 frames.(our assumption). But the frames 252 and 257 will fail in pass one of wipe detection algorithm in the condition $count > 2$ as these are not the first and fifth frames of the wipe. So it is very important in our algorithm that we track the first frame in a wipe, to decide whether there is wipe starting with this frame or not. Hence one more module is added in our algorithm, called as *Local Window of Poor Correlations Detection Algorithm (LWPC)*. This algorithm will search for all the frames before and after the present frame indexed by f such that their five frame correlations are below the threshold T_{lw} . *While searching in both directions (forward or backward), the search stops once a frame with its five frame correlation above T_{lw} is reached.* T_{lw} is typically chosen as 0.42. In general it can be less than or equal to T_{main} . The starting and ending frames in

this window are named as *wbegin* and *wend* , and these values are used by wipe and dissolve detection algorithms.

2. **In Dissolve Detection:** Let us consider that a dissolve is between 200 to 205 frames. Suppose f points to 196, and we calculate five frame correlation between 196 and 201 frames; this correlation will be generally just above T_{main} , since dissolve is gradual transition, and only dissolve frame is present. Hence search starts for next transition between 201 and 206. Now dissolve will get detected but actual dissolve is from 200 th frame. In such a case also, to track exact starting frame in gradual transitions, the LWPC algorithm is needed.

Fig 3.6 represents the flowchart for LWPC Detection Algorithm. From the flowchart, it is clear that, we first find poor correlations in backward direction, and then in the forward direction. Then we designate the boundary frames by *wbegin* & *wend* and return these values.

3.4.6 DISSOLVE Detection Algorithm

Fig 3.8 represents the flowchart of Dissolve Detection Algorithm.

This algorithm is invoked from the main shot detection algorithm, whenever the five frame correlation between frames f and $f + 5$, i.e., $corr5 \leq T_{main}$, and there is no other type of transition found. Detection of dissolves is a difficult task, because there are many types of changes taking place within few frames, which may be detected as Dissolves. For example, some player comes in between a different shot and moves slowly within few frames say 3 to 8, from the shot. Such things will be detected as dissolves. Our algorithm for Dissolve detection is designed, keeping in view the definition of dissolve, which is mentioned in section 3.4.1 and as well from fig. 3.2 (c). The aim is to find whether there is a dissolve between frames bounded by *wbegin* & *wend* . This is explained below:

1. **First Condition:** Firstly we check whether the frames *wbegin* & *wend* satisfy the conditions that $wend > (wbegin + 5)$ and $wend < (wbegin + 12)$. This will ensure reduction number of false positives for dissolves that arise from reasons

as explained above. These conditions are based on the assumption the dissolve will take place within 3 to 7 frames, after carefully examining the correlation statistics. This algorithm is not robust. If any of these two conditional statements fail, then the frame index parameter f is modified to $f + 5$ so as to start search for next shot transition.

2. **Second Condition:** Once step 1 above is passed, then it indicates that there is possibility of dissolve from $wbegin + 5$ to $wbegin + 12$. But there can be false positives arising from reasons mentioned above, that will last for same duration as dissolves, i.e., 3 to 7 frames. To reduce them, we check the five frame correlations between first and fifth frames, and five frame correlations between frames just following the dissolve. For an actual dissolve, in the first case we get a poor correlation, and in the second case a good correlation as these frames belong to same shot. This will reduce the false positives, as the algorithm is close to definition of dissolve.

One additional block is included in our Dissolve detection algorithm which we named as *Check Cuts from Dissolves Algorithm*. This algorithm is invoked in dissolve detection algorithm when we find that first condition is satisfied and second condition in dissolve detection is not satisfied. We declare that there is no dissolve, and we now check for any missed cuts within frames $f + 5$ and $wend + 1$.

The reason for this block is best illustrated by an example:

Consider that our five frame correlation in main algorithm of shot detection, $corr5$ to be less than T_{main} such that we check for any shot transition. And we have passed through the Cut Detection and Wipe Detection algorithms, and found there is no cut or wipe. Now the algorithm transfers control to dissolve detection algorithm where, second condition is not satisfied. Let f point to frame no 300, and window boundaries, $wbegin$ & $wend$ be 297 and 308. Let us assume that there is a cut from 308 to 309 frames. Now since all types of transition detections fail, if f is modified to 309, then we miss this cut. So even if there is no gradual transition within window frames, there can

be a cut, as our earlier cut detection algorithm checks for cuts only from f to $f+5$ frames, and we did not check for cuts from $f+5$ to $wend+1$ frames earlier.

3.4 Performance Criteria

In information retrieval applications, there are several methods to measure the performance of the systems. One popular method [10] is to use the Precision (P) and Recall(R) values. These are expressed as follows:-

$$P = \frac{NC}{(NC + FP)} \quad \text{and} \quad R = \frac{NC}{(NC + FN)}$$

Where NC = No of Correct Detections

FP = No of False Positives

FN = No of False Negatives

By giving equal weights to Precision and Recall, we can derive another metric F to measure the overall system performance. It is given by:

$$F = \frac{2 * R * P}{(R + P)}$$

3.5 Experimental Results

In this section we present experimental results of our Shot Segmentation Algorithm for three video clips. The experimental environment is as below:

- System Details : Intel 845 GLLY, 1.6 GHz Clock Speed, 256 MB RAM, 512 KB Cache Memory.
- Operating System: Windows XP Professional.
- Platform: Microsoft Visual C++ with *Intel® Open Source Computer Vision Library*.
-

The video clips considered are as follows:

1. Video Clip 1:

- Name: Women Cricket Match.
- Type : One Day International Match.
- Duration: 26 min 31 sec 66 msec
- Frame Rate: 12
- Total No of Frames: 19100

2. Video Clip 2:

- Name: Men Cricket Match.
- Type: Test Match.
- Duration: 42 min 49 sec 74 msec
- Frame Rate: 12
- Total No of Frames: 30837

3. Video Clip 3:

- Name: Men Cricket Match
- Type: Test Match
- Duration: 44 min 48 sec 23 msec
- Frame Rate: 12
- Total No of Frames: 32259

The size of the frame in all the above video inputs is 352x288. Now we present the output file generated by our algorithm, for the first 1000 frames of video clip 2 referred above.

3.6.1 Sample Shot Data File for first 1000 frames of Video Clip 1

shot no	from frame	length	Transition to Next Shot
1	0	16	CUT
2	16	64	CUT
3	80	13	CUT
4	93	23	CUT
5	116	19	CUT
6	135	26	CUT
7	161	8	CUT
8	169	22	CUT
9	191	34	WIPE
10	235	105	DISSOLVE
11	345	12	CUT
12	357	2	CUT
13	359	81	WIPE
14	450	136	CUT
15	586	50	CUT
16	636	28	CUT
17	664	65	CUT
18	729	72	CUT
19	801	18	CUT
20	819	107	CUT
21	926	3	CUT
22	929	18	CUT
23	947	109	CUT

3.6.2 Shot Detection Results

1) Video Clip 1: One Day International Women Cricket Match

No of frames : 19100

Frame Rate : 12

Duration of Clip : 26 min 31sec 66msec.

Run Time : Around 4min.

	CUTS	WIPES	DISSOLVES	SHOTS
CORRECT DETECTIONS	210	13	0	223
FALSE POSITIVES	28**	0	6***	28
FALSE NEGATIVES	0	4	0	4
PRECISION	88.23%	100%	0%	88.85%
RECALL	100%	76.47%	NA	98.24%
OVERALL PERFORMANCE	93.75%	86.67%	NA	93.17%

Table 3.1 Shot detection results for women cricket match

** All False Positives for Cuts are Camera Panning detections.

*** All Detected dissolves are False Positives, as no Dissolve is used in Video.

2) Video Clip 2: Men Cricket Match (Test Match)-I

No of frames : 30837

Frame Rate : 12

Duration of Clip : 42 min 49sec 74msec.

Run Time : Around 6min.

	CUTS	WIPES	DISSOLVES	SHOTS
CORRECT DETECTIONS	252	51	46	355
FALSE POSITIVES	13	1	19	30
FALSE NEGATIVES	18	13	25	51
PRECISION	95.09%	98.07%	70.76%	92.21%
RECALL	93.33%	79.68%	64.79%	87.44%
OVERALL PERFORMANCE	94.20%	82.82%	67.64%	89.76%

Table 3.2 Shot detection results for Men's cricket match I

2) Video Clip 3: Men Cricket Match (Test Match)-II

No of frames : 32259

Frame Rate : 12

Duration of Clip : 44 min 48sec 23msec.

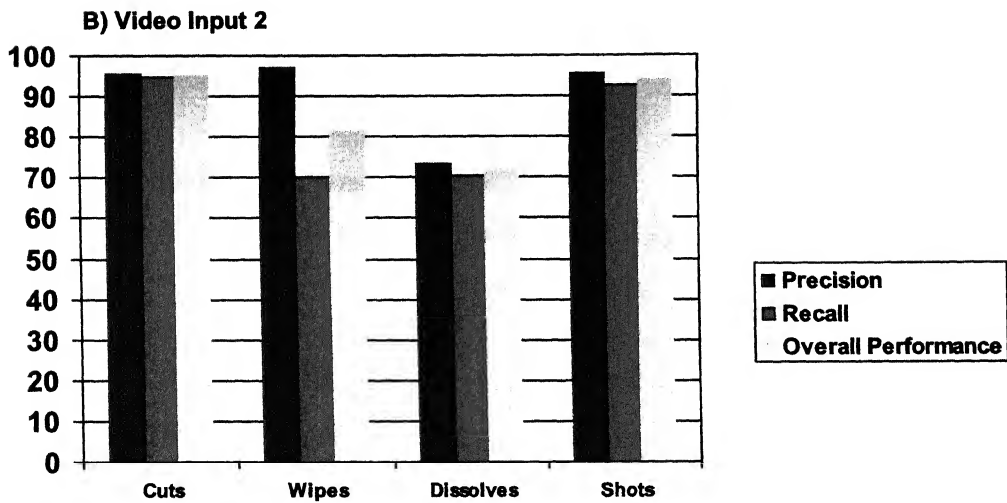
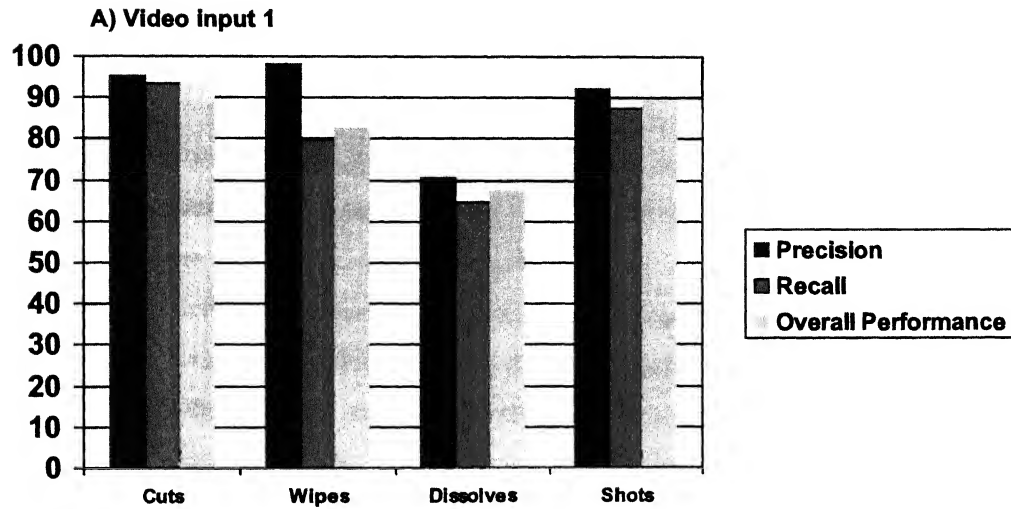
Run Time : Around 7min

	CUTS	WIPES	DISSOLVES	SHOTS
CORRECT DETECTIONS	266	35	36	348
FALSE POSITIVES	12	1	13	16
FALSE NEGATIVES	15	15	18	28
PRECISION	95.68%	97.22%	73.47%	95.60%
RECALL	94.66%	70.00%	70.58%	92.55%
OVERALL PERFORMANCE	95.17%	81.39%	71.99%	94.05%

Table 3.3 Shot detection results for Men's cricket match II

The shot detection results of Men's cricket match are shown by bar diagrams in Fig 3.9 (A) and (B)

FIG 3.9 SHOT DETECTION RESULTS
(Men's Test Cricket Match)



3.7 Performance Evaluation and Computational Complexity

3.7.1 Performance Evaluation

Once shot detection results are obtained, we were naturally interested to know how our algorithm performs in comparison to existing approaches. For this, we referred one recent good reference on performance comparison of shot change detection methods [10]. In this paper, the authors have considered many shot detection methods, which are based on color histograms, features computed from block motion or compression parameters to compute the frame difference. The comparison was done for almost all the histogram metrics that were discussed in section 3.3.2, for different color spaces and dimension. The outcome of performance comparison for shot detection methods using color histograms is that when Precision is at 95%, all the existing histogram methods showed a recall of less than 65%. This will amount to the overall system performance (from 3.5) of 77.18% and these results are for Cut Detection. Here the data set consisted of extensive graphical effects and motion. The aim of this paper was to propose a precision value of 90% and a recall value of 70% and the algorithm should be robust.

Now if we look at the shot segmentation results of our algorithm give in 3.6.2, then we got precision and recall values around 88 % and above for cuts, and for gradual transitions the precision is over 97% and recall over 70% for wipes. But for wipes the precision and recall values are above 65%, which is also a reasonably decent figure.

3.7.2 Computational Complexity

Though we thought to compare our algorithm with existing techniques in terms of computational complexity, but the problem is that the due to different systems and operating environments and as well the difference in the data that is analyzed we could not proceed further with this idea. In contrast to this, we have simple and powerful arguments which justify that our shot detection algorithm is computationally efficient and robust with respect to type of cricket match and lighting conditions. On observing the shot detection results in 3.6.2, it is clear that our shot detection algorithm is completing its task in around 6 min for a 40 min video. This performance is very much useful when

we try to complete annotation of cricket video of 40 min duration in less than 15 min.

The reasons for our computational efficiency are:

- We have considered five frame correlations for our criteria of deciding whether there is a shot transition or not (i.e., $corr5 \leq T_{main}$ where T_{main} is around 0.42). Hence if this condition is not satisfied then the algorithm searches for shot transition in next five frames. The importance of this step is that in a video clip of length say 30000 frames, it is observed there will be typically around 500 transitions and only 750 five frame correlations satisfy the above condition. This implies that we are reducing computational cost by 40 times. This technique has not been used so far.
- Our algorithm for detection of wipes or dissolves involves evaluating only two five frame correlations, while existing approaches use cumulative correlation of successive frames and twin threshold techniques which is computationally very expensive.

CHAPTER 4

ANALYSIS OF CRICKET VIDEOS

4.1 Introduction

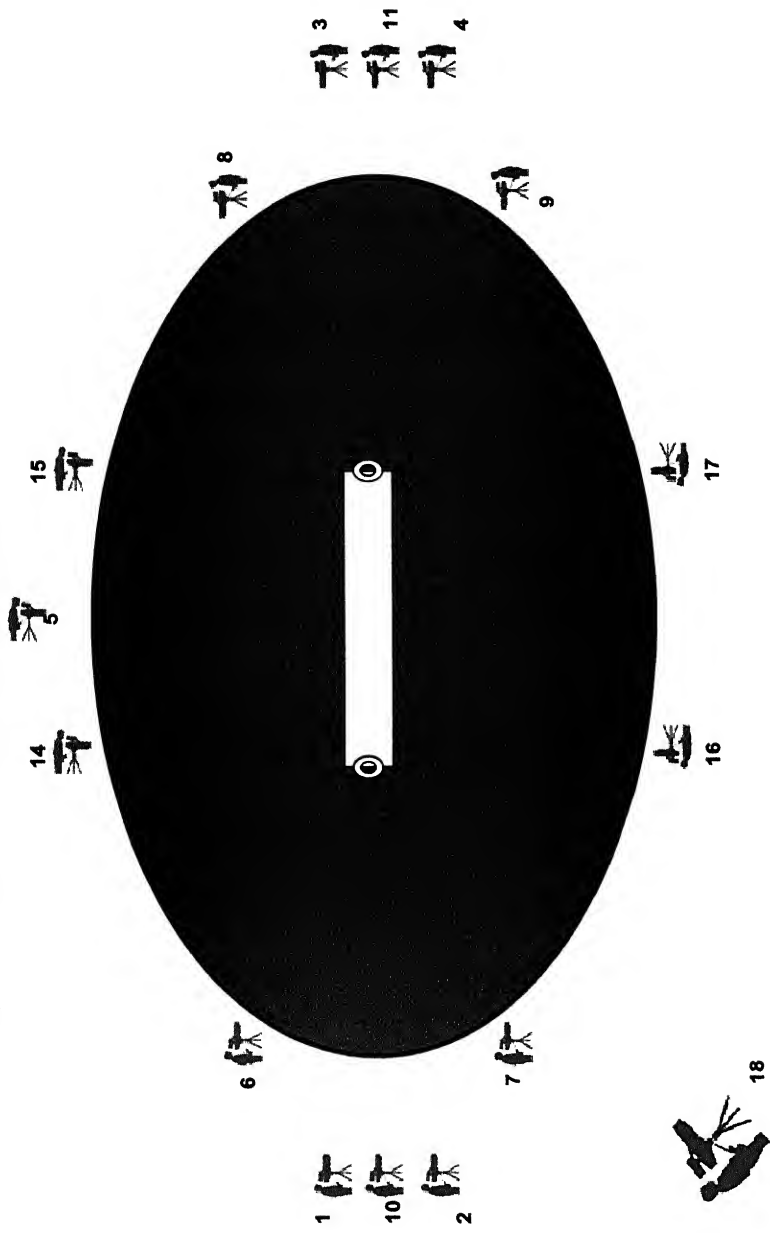
By analysis of cricket videos, we mean understanding the domain knowledge of cricket videos from the automatic annotation point of view. This implies that one needs to know about various kinds of cameras used, shots present, shot sequences, highlight actions, etc regarding the sport concerned in order to facilitate the automatic annotation task. Also in this chapter, we will identify the structure and complexity of cricket videos. We make use of this domain knowledge to demonstrate that it is possible to generate meaningful semantic annotations applicable to the domain, in chapter 5.

4.2 Typical Positions of Cameras

Since cricket involves 15 people on the field (eleven players of fielding side, two batsmen from Batting side, along with two umpires on the field), hence many cameras are required to shoot a live coverage of cricket video. Fig 4.1 shows the typical camera positions used in recent one day international cricket match. Altogether there are 18 cameras used and the purpose of all the cameras are explained below:

<i>S.No</i>	<i>Purpose of Camera</i>
1 & 4	Pitch Camera
2 & 3	Ball following from Pitch
5	Mid-Wicket Camera
6, 7, 8, 9	Batsman Close-up, Bowler Run-up and Ball following
10 & 11	Super Slow Motion Cameras.(Covers from Bowler's Bowling action and further)
12 & 13	Stump Vision Cameras
14, 15, 16, 17	Run Out Cameras
18	Bird View Camera

Fig. 4.1 Typical Camera Positions in a Cricket Telecast



Courtesy: DoorDarshan Sources

These cameras are further grouped, to facilitate easy identification of the shots at semantic level, as given below:

<i>S.No</i>	<i>Name of Group</i>	<i>Cameras</i>
1.	C1	1, 4
2.	C2	2, 3
3.	C3	5
4.	C4	6, 7, 8 and 9
5.	C5	10, 11
6.	C6	12, 13
7.	C7	14, 15, 16, and 17
8.	C8	18

Table 4.1 given below shows the various shots in a cricket video and the camera groups used to televise them.

1	<i>BATSMAN_CLOSE</i>	<i>C4</i>
2	<i>BOWLER_CLOSE</i>	<i>C4</i>
3	<i>BOWL_BAT</i>	<i>C4,C5</i>
4	<i>BOWLING</i>	<i>C4,C5</i>
5	<i>BATTING</i>	<i>C4,C5</i>
6	<i>FIELDING</i>	<i>C3,C4,C7</i>
7	<i>IDLE/PLAYER</i>	<i>C4,C7</i>
8	<i>AFTER_FIELD_PITCH</i>	<i>C4,C7</i>
9	<i>AUDIENCE</i>	<i>EXCEPT C6</i>
10	<i>UMPIRE</i>	<i>C4</i>
11	<i>COMMENTATORS BOX</i>	<i>C8</i>
12	<i>SLOW_MOTION_REPLAY</i>	<i>C5,C6,C7</i>
13	<i>MISCELLANEOUS</i>	<i>ALL</i>

Table 4.1 Various shots vs. Camera groups used to televise them

The different shots referred in above table are explained in next section.

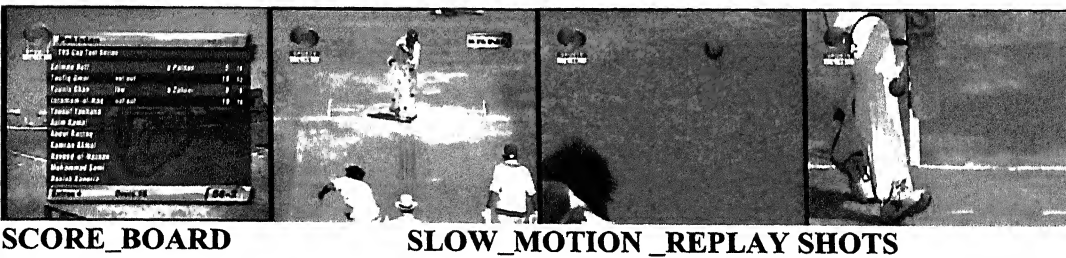
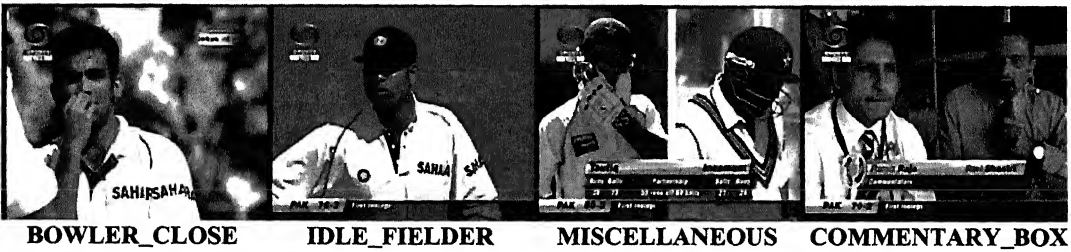
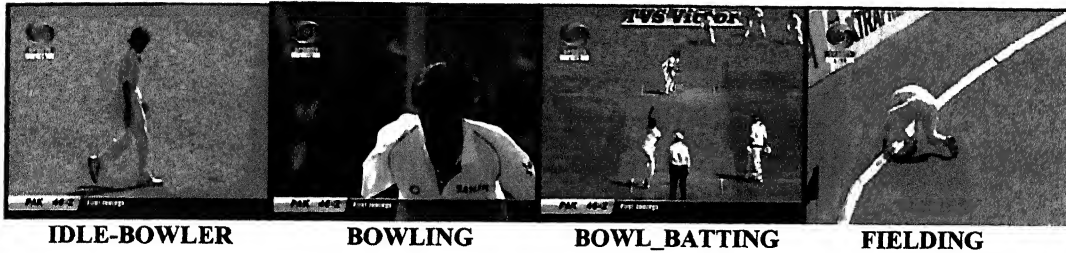
4.3 Semantic Classification of Shots

Due to many number of cameras used, and as well due to the lot of sporting action involved in a cricket match i.e., for almost 6 hours/day; we get a wide variety of shots.

Fig 4.2 shows various shots classified based on the viewer level semantic information.

1. **BATSMAN-CLOSE:** Batsman is shown in a close shot. Generally occurs before a ball is bowled but not restricted to this only.
2. **BOWLER-CLOSE:** Similar to type 1, but Bowler is shown; the difference from type 1 is that here there is no Helmet, Bat.
3. **BOWL_BAT:** In this shot, Bowling action of Bowler and Batting Action of Batsman until hit or miss occurs, is covered. In addition this shot can have either Fielding or Keeping actions included in them. The difference in Fielding or Keeping actions is the length of shot.
4. **BOWLING:** This covers only bowling action of Bowler run-up.
5. **BATTING:** Covers only Batting Action. Occurs very rarely, as on almost all occasions type 3 shots covers this action.
6. **FIELDING:** Covers only field and the fielder(s) when the ball is hit.
7. **IDLE_FIELDER/BOWLER:** This covers a fielder or a Bowler. Generally less motion is present in such shots.
8. **AFTER_FIELD_PITCH:** This shot covers the pitch, after a ball is thrown by a Fielder to Wicket Keeper, or Bowling end.
9. **AUDIENCE:** This shot covers audience whenever there is a highlight action like a boundary, a sixer or a wicket etc. These shots can occur anywhere in Video.
10. **UMPIRE:** Generally occurs whenever there is any highlight action, after either of shots 3, 5, 6, or 8.
11. **COMMENTATORS BOX:** Shows commentators, either single person or two. Very rare occurrence.
12. **SLOW_MOTION_SHOTS:** Whenever a highlight action occurs, replay of previous event sequence takes place with a wipe action before and after this shot.

Fig 4.2 TYPICAL SHOTS IN CRICKET VIDEOS



13. **MISCELLANEOUS:** In addition to above commonly occurring shots, there are many other shots which occur in cricket videos; these shots occur at random. Some of them are given below:

- Batsman walking over the pitch in between transition of two successive overs.
- Appeal of Bowler to Umpire.
- Two Bat Close shots merged into a single shot with a longitudinal wipe.
- Players' Gallery.
- Local weather conditions, tourist places of interest of the place where the match is played.

Apart from all the above shots, some shots are shot by different cameras and have different low level information of video. Some of them are given below in Figs 4.3 (a) and (b).

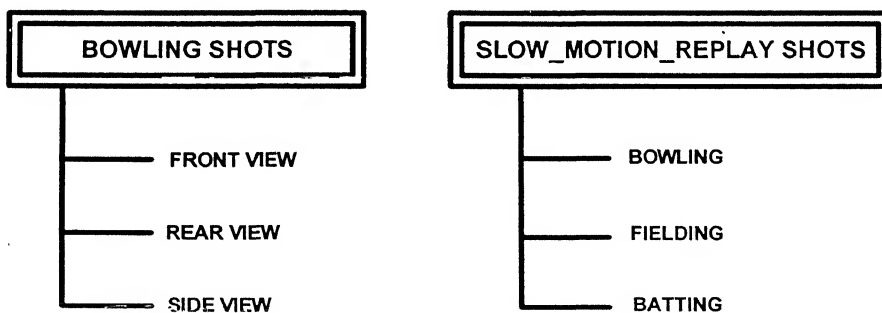


Fig 4.3 (a) and (b) Sub-Classification of Shots

4.4 Complexity of Domain Knowledge

In this section, we present how complex the domain knowledge of cricket videos is, in spite of cricket being a highly structured game, from annotation point of view. Even though the higher level semantics do not change; but the low level content like color information, illumination conditions, motion etc, change very much from one cricket match to other due to change in type of match, telecasting thumb rules of cameramen etc. To be more specific, in test cricket matches players wear white uniform, while in one day internationals the players wear color uniform that depends on their country. One another difference between one day and test matches is that, some times one day matches will be played in the night also, thus causing some shots like audience to have low illumination. Even if we consider only one type of matches say test matches, then also due to different people doing coverage for different matches, some new type of shots may result.

The problems does not end here, as even within the same match, there will be lot of content variation within a shot due to zooming of camera, and the ideal annotation tool should consider the zoomed frames to be within the same shot. Content variation is also present across shots, i.e., no two shots of same type will have same information; they may differ in length, and player i.e., a Batsman Close shot will have at most eleven Batsmen shown in a match. To add to this, if the number of cameras is more, then there will a wide variety of shots, and content variation across shots is more prominent as for example same bowling action may be shot in three different views viz., front, rear and side. Hence an ideal annotation system should address all the above issues.

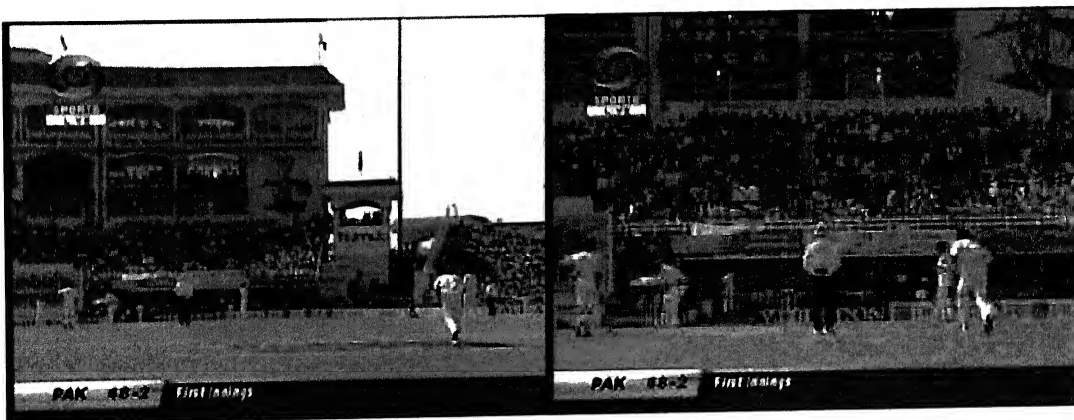
The above complexities in domain knowledge are shown in figs. 4.4 to 4.6. The content variation within a shot is shown in Fig 4.4, for three different types of shots and the content variation across shots is shown in Fig 4.5. The distribution of various shots for a men cricket match video clip of length around 40 min is analyzed and represented in pie diagram in Fig 4.6.

Fig. 4.4 CONTENT VARIATION WITHIN A SHOT

A) BATSMAN_CLOSE SHOT



B) BOWLING SHOT

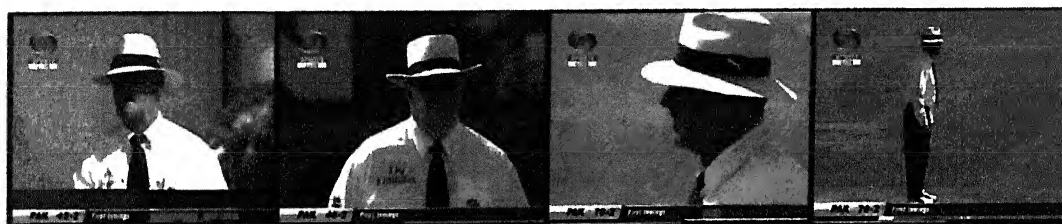


C) FIELDING SHOT



Fig 4.5 CONTENT VARIATION ACROSS SHOTS HAVING SAME SEMANTIC MEANING

A) UMPIRE SHOTS



B) BOWLING SHOTS



C) AUDIENCE SHOTS

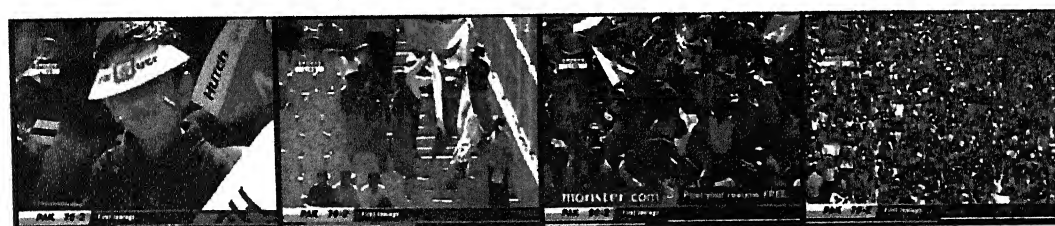
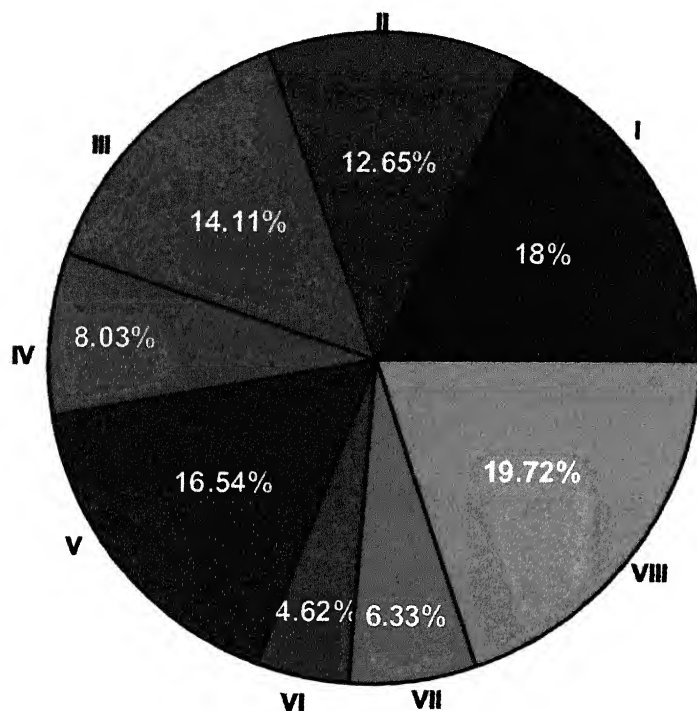


Fig 4.6 TYPICAL DISTRIBUTION OF VARIOUS SHOTS IN A CRICKET VIDEO CLIP



NOTE:

- I : BATSMAN_CLOSE
- II : BOWLER_CLOSE+FIELDER_CLOSE
- III : AUDIENCE+UMPIRE+SCOREBOARD
- IV : SLOW_MOTION_REPLAY
- V : BOW_BAT
- VI : FIELDING ONLY
- VII : BOWLING ONLY
- VIII : MISCELLANEOUS

CHAPTER 5

AUTOMATIC SHOT CLASSIFICATION AND ANNOTATION

5.1 Introduction

This chapter aims at presenting our shot classification algorithm based on color histograms motion templates derived from shot energy. Also we present our algorithm to detect slow motion replay shots based on the shot energies derived from the difference images of consecutive frames.

The basic tasks in video annotation are given below:

1. Temporal segmentation of video stream into shots.
2. Detection and recognition of text appearing in captions.
3. Extraction and interpretation of audio track including speech recognition.
4. Automatic classification of shots.
5. Visual summarization of shot content.
6. Semantic annotation.

In general, a bottom-up approach, moving from low level perceptual features to high level semantic descriptions is followed. Our focus has been on tasks 1,4, 5 and 6. Towards this aim, the first step i.e., temporal segmentation of data, is given in chapter 3 along with performance of our own algorithm. Once the task of shot segmentation is performed, the next step in automatic annotation is to classify these shots into various categories using the domain knowledge as explained in chapter 4. We have classified shots into semantic categories, based on the low level image features like color histograms, and an energy metric derived from difference images of consecutive frames. This energy metric in a simple way represents the amount of motion present in a shot, which is used to identify slow motion replay shots; which can be used for highlights generation. After classifying the shots automatically, this information can be used at various levels to generate annotations that are based on higher level semantics. Our annotation has been based on application context, keeping in view the requirements of users and Broadcasters. The Broadcasters can use the annotated videos for

posterity/production logging, or the users on worldwide web accessing the selected segments of video. We will present these applications in section 5.6.

5.2 Existing Work

As far as the research work regarding annotation of sports videos are concerned, many researchers have worked mainly sports like Soccer, Tennis, and Basketball etc [1][21][22][23][24][26]. Very few researchers have worked in analysis and annotation of cricket videos [2][20][25]. In [2] a dynamic programming based HMM model is used to analyze cricket video sequences to facilitate highlight generation and content based video retrieval. The authors classified real cricket video sequences into four semantically meaningful classes like 'Ball Movement', 'Fielding', 'Pitch Action'(Balling and Batting), and 'Wicket Keeping'. The HMM framework is used by them for both shot detection and classification purposes. Our approach is also to apply HMM modeling, but at the shot classification level, with an aim to have more classes as far as possible so as to develop more informative annotation. The present work has been confined up to shot classification into more categories, based on color information and an energy metric which is used to detect slow motion replays.

Since we have also attempted the problem of detection of slow motion replays, which is very much essential for highlights generation, we present some of the existing works here. Many researchers have worked on this problem by making use of human generated editing effects, from detecting logos at scene transitions, and motion based features for detecting slow motion shots. In [25] the authors used motion metrics, based on frame difference methods, motion compensated block based frame difference methods etc to identify a slow motion shot.

Our approach has been to use both editing effects, and frame difference metrics, to evaluate the energy of a shot, which will give an idea of whether there is motion or no motion. But this technique cannot be used to classify shots, as it needs more robust motion estimation algorithms, using optical flows, or any other methods.

5.3 Automatic Shot Classification

5.3.1 Shot Representation

To classify shots automatically, several low level visual primitives like image edges, corners, segments, curves, color histograms, optical flow, textures, shape, length of the shot etc., have to be first found out for every type of shot. Even transition probabilities can form a feature vector for each type of shot. After all features are extracted, then we represent the contents of each shot using a *color histogram vector* and a *feature vector*. The histogram vector is used to match the content of a shot with the representative shot of certain categories, while the feature vector can be used by a Decision Tree to categorize the shots into one of the remaining categories.

In our work, the color histogram vector for a particular shot is obtained from the normalized cumulative histogram of all shots of that type; and as part of the feature vector, we have found an estimate of motion in the shot from the energy of the difference images; where the difference image is obtained from pixel wise difference of two consecutive images.

5.3.2 Extraction of Shot Templates

Any automatic shot classification algorithm should first extract the color and feature vectors of each type of shot, and use them as templates with different weights associated for each feature for different kinds of shots. So the first step in shot classification is the extraction of color and feature templates. We have implemented algorithm to classify the shots by extracting three different templates, which are presented below:

- *Image-wise Color Templates:* In this method, we first extract cumulative histograms of shots of a particular type and then normalized to get a single template representing a shot category. This is done for all the shot categories that are to be considered. Keeping in view the content variation across shots, which we have discussed in section 4.4, some types of shots like Bowling have more than one template.

- *Block-wise Color Templates:* Here we extract templates as discussed in image wise templates method discussed above, the only difference being that now instead of one template, we get N templates, where N represents the number of blocks the frame is divided into. The advantage of this method is that spatial variations in shot content are taken care of here. We have computed block templates for the case of $N=4$.
- *Motion templates from shot energy:* Since sports videos have lot of sporting action in them, any feature that corresponds to motion will be useful. In this method firstly we compute the difference image from two consecutive frames as given below:

Consider a frame of size $M \times N$ pixels, and there are L frames in a shot.

Let D_i be the difference image, and $F_E(i)$ be frame energy of difference image D_i .

We have the following equations:

$$D_i(x, y) = f_i(x, y) - f_{i-1}(x, y) \quad \forall i = 1 \text{ to } L-1$$

$$\text{where } x \in \{0, 1, \dots, M-1\}, y \in \{0, 1, \dots, N-1\}$$

$$F_E(i) = \frac{1}{MN} \sum_{\forall x, y} D_i^2(x, y)$$

Now the shot energy S_E is found by,

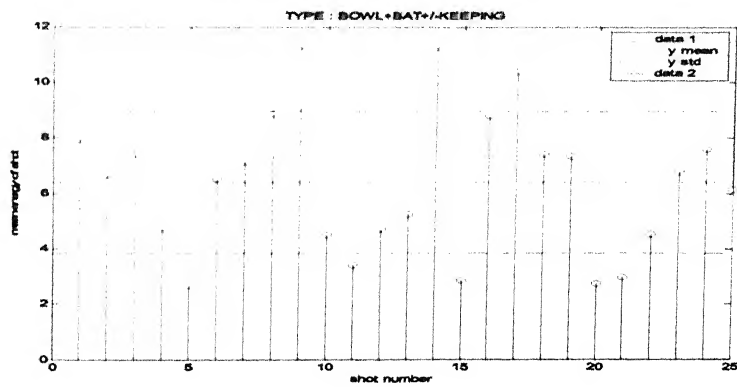
$$S_E = \frac{1}{L-1} \sum_{i=1}^{L-1} F_E(i)$$

This shot energy is computed for all shots of a particular type, and a super mean and super variance values of shot energies are evaluated. These super mean and variances are used as shot templates. Fig 5.1 (a) to (d) shows shot energy profiles of four different categories of shots, i.e., they are plots of S_E vs shot number for a particular type of shot. From every plot we find the super mean and variance denoted by $s_mean(i)$ and $s_var(i)$ which act as the motion templates for an i^{th} shot category.

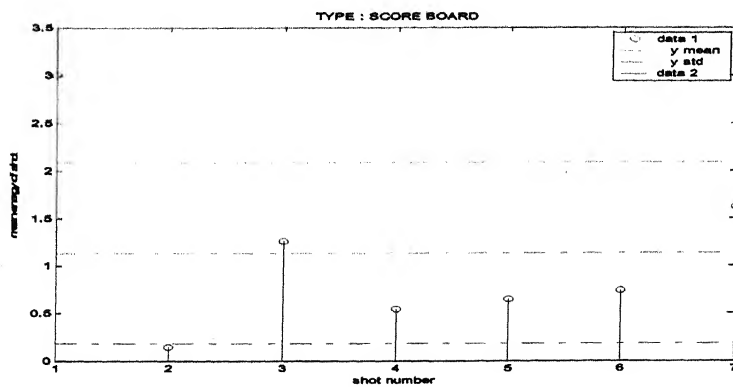
Once the shot templates, be it color or motion, are extracted, then these are used in Shot Classification Algorithm. We present the algorithm which uses the shot templates explained above, along with their performance comparison in the following sections.

Fig 5.1 Plots of mean shot energy vs. shot number for different categories of shots

A) BOWL+BAT+KEEPING SHOTS



B) SCORE BOARD SHOTS



C) SLOW MOTION REPLAY SHOTS

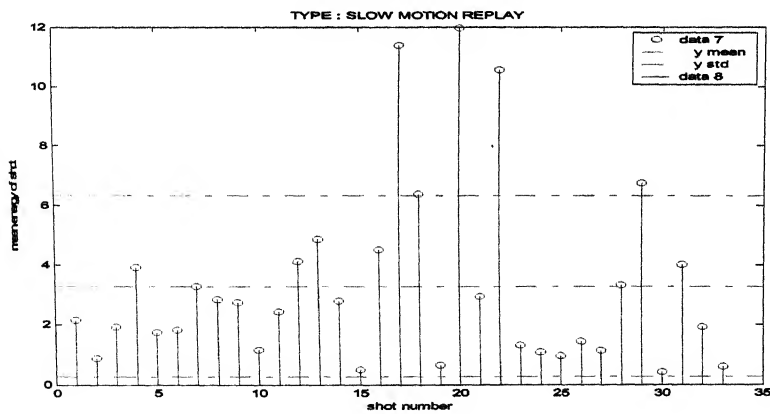
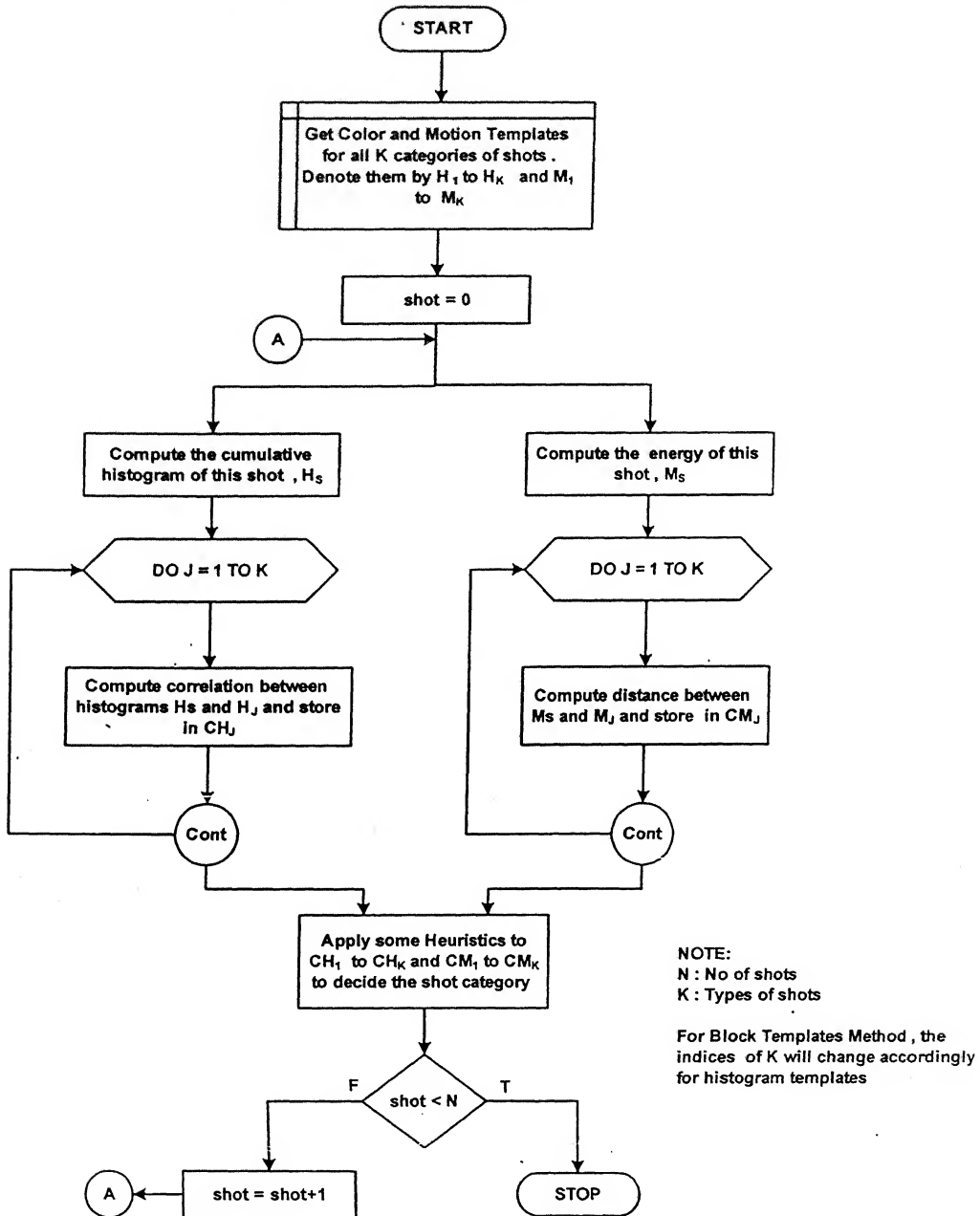


Fig 5.2 SHOT CLASSIFICATION ALGORITHM



5.3.3 Shot Classification Algorithm

Fig 5.2 shows the flowchart of generalized shot classification algorithm, which accepts shot templates and evaluates type of the shots by different approaches as explained below:

(a) Using Color Templates Only:

This algorithm makes use of the different shot templates that are generated as explained in section 5.3.2. Once we have the shot templates, then for every shot, the shot classification algorithm computes how close this shot is to all the shot templates. The distance metric in case of color templates is the *3D Histogram correlation*. Firstly we compute the cumulative histogram of the shot to be classified. In the case of image templates method, we get one correlation value for every comparison of shot histogram and template histogram, whereas for block templates method, we get N correlation values; N being the number of blocks. This process is repeated for every shot type, so that we get K correlation values for K shot types in case of image templates method, and $4 * K$ correlation values in case of block templates method. Now based on these correlation values and using any heuristic the shot type is decided. The heuristic used by us is simple. First we find how many block templates have correlation greater than 0.65 and then decide the shot type based on the count value. The shot classification results for both image and block templates methods is presented in next section.

(b) Using Motion Templates Only:

In the case of motion templates method, at first, we compute the shot energy S_E of the shot n to be classified and then check whether this value is within the energy statistics of a particular shot type say k , we check the condition, if $|S_E - s_mean(k)| \leq s_var(k)$, where $s_mean(k)$ and $s_var(k)$ are the mean and variances of energies obtained from plot of shot energies of type k (Some examples of such plots were shown in Fig 5.1). If the above condition is satisfied, then the shot n is classified to be of type k . Though this method sounds good, but when we observed the energy plots of all the shot types, it is found that the motion templates cannot be fully relied on to classify the shots. The

reasons for this is that, since in cricket videos like any sports videos, there is motion in many types of shots which will make it difficult to draw fine boundaries between different categories of shots. Another reason is that we have only mean and variances of energies, which are not sufficient statistics. Also since this method is computationally expensive, so we did not attempt this method for shot classification.

(c) Using Both Color and Motion Templates: In this method, we use both block color templates and motion templates, to determine the type of the shot using some heuristics. From comparison with block color templates, we get correlation vector for each shot type, and on comparing with motion templates, we find whether the shot to be classified as type k , say, has energy satisfying the condition that $|S_E - s_mean(k)| \leq s_var(k)$.

Thus this is a two step process, in the first pass, we check color histograms, and then pass on to second stage where we will check shot energies.

We present in subsequent sections, the results of our automatic shot classification algorithm using all the three different approaches explained above.

5.3.4 Shot Classification Results

In this section we present results of our automatic shot classification algorithm using three different combinations of shot templates as explained in section 5.3.3 above.

Table 5.1 given below is the exhaustive list of all the results, and fig 5.3 shows a simplified view of results of overall system performance for important shot categories.

NOTE:

1. P-I, R-I and OSP-I: Precision, Recall and Overall System Performance of Shot Classification Algorithm using *Image wise Color Templates*.

2. P-II, R-II, and OSP-II: Precision, Recall and Overall System Performance of Shot Classification Algorithm using *Block Color Templates*.

3. P-III, R-III, and OSP-III: Precision, Recall and Overall System Performance of Shot Classification Algorithm using *Both Block Color and Motion Templates*.

Type of Shot	P-I	R-I	OSP-I	P-II	R-II	OSP-II	P-III	R-III	OSP-III
<i>BAT CLOSE</i>	60.2	75.67	67.06	79.36	67.56	72.99	47.69	83.78	60.78
<i>AUDIENCE</i>	76.1	64.0	69.56	50.0	60.0	54.55	35.13	52	41.93
<i>BOWLER CLOSE</i>	55.0	42.31	47.83	29.41	76.92	42.55	50.0	50.0	50.0
<i>UMPIRE</i>	66.6	23.08	32.29	77.77	26.92	39.99	72.72	30.77	43.24
<i>SCORE BOARD</i>	100	71.4	83.31	83.33	71.43	76.92	100.0	57.14	72.76
<i>BOWL+BAT</i>	36.7	52.94	43.37	44.7	58.46	50.66	48.1	55.88	51.42
<i>FIELDING ONLY</i>	66.6	21.05	31.99	33.33	15.79	21.43	37.5	15.79	22.22
<i>BOWLING ONLY</i>	16.6	11.54	13.64	38.19	61.54	47.13	34.48	41.67	37.73
<i>FIELDER CLOSE</i>	42.8	32.58	37.2	56.25	47.37	51.43	71.43	26.31	38.45
<i>SLOW MOTION REPLAY</i>	80.0	12.12	21.05	71.43	15.15	24.99	60.0	9.1	15.8
<i>TOTAL SHOTS</i>	50.8	46.66	48.67	50.91	51.98	51.44	47.49	49.84	48.64

Table 5.1 Shot Classification Results

5.3.5 Results : Comments

From the shot classification results, as presented in table 5.1 and in fig 5.3, it is clear that whatever be the shot templates that are selected, be it image wise color templates, block color templates or using both block color templates and motion templates, the overall system performance for classification of shots is around 50%. Since we have used only two low level features, the performance is as expected. It is important to note from the results that, the block color templates method perform better in terms of overall system performance for many of the shot categories when compared to other two methods. The third method, i.e., which uses both block color templates and motion templates should have performed even better. But due to the reason that we are not utilizing the energy profiles, but only the mean and variances, so it is not performing to its best.

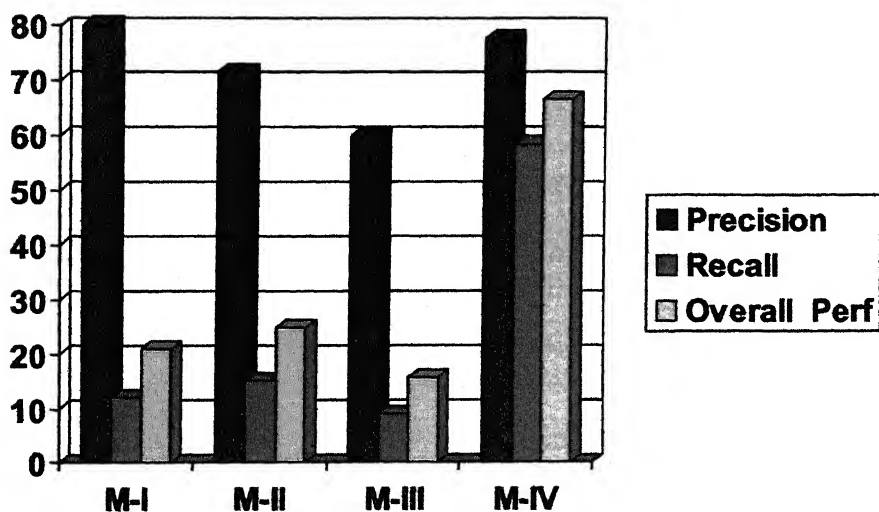
5.4 Detection of Slow Motion Replays

On observing the shot classification results of our algorithm, it is clear that two categories of shots i.e., Fielding and Slow Motion Replays have very less overall system performance. Out of these two types of shots, slow motion replays are more in number and are very important as far as highlights detection is concerned. Because in cricket videos, any highlight action (by highlight action we mean a boundary, sixer, wicket or any action which has high semantic significance) will be shown at least once again, so that the Television viewers do not miss important events. To differentiate between a replay of earlier event with the current event, in cricket broadcasts, wipes are used as start and end transitions of replay events. This is a commonly used technique, though it is not followed universally. In the video data we analyzed, almost all the highlight actions are bounded by wipes. It is important to note that every slow motion shot is a replay type shot but the converse is not true.

We have developed an algorithm called *Wipes and Motion Templates Method* to detect slow motion replay shots. As the name suggests the slow motion shots are identified by the help of wipes and motion templates. The algorithm is like this:

First we see for all shots such that there are two wipe transitions within some temporal length, say 400 frames; then we find the energies of shots within these transitions, and if the energy of any shot is close to energy of the slow motion type shots, then that shot is declared to be a slow motion replay shot. Using this simple technique, we have detected slow motion replays with an overall system performance of about 66.7%. The results for detection of slow motion replays are shown in fig. 5.4.

Fig 5.4 Shot Classification Results for Slow Motion Replay Shots



Note:

M-I : IMAGE WISE COLOR TEMPLATES METHOD

M-II : BLOCK COLOR TEMPLATES METHOD

M-III: BOTH BLOCK COLOR AND MOTION TEMPLATES METHOD

M-IV : WIPES AND MOTION TEMPLATES METHOD

5.5 Annotation Cues

The third and final step in automatic annotation is extracting semantic annotations based on the classified shots. The efficiency of any retrieval system depends mainly on effective annotation and summarization. Some of the examples of annotation cues and how they are generated are given below:

1. How many balls are bowled?..... From no of batting shots
2. How many fours are hit?..... From no of Boundary shots
3. Whether OUT or not? From Stump Vision Camera o/p
4. Match Highlights? From Wipes & Slow Motion shots
5. Who is Batting or Bowling? From Face Detection Alg &
audio/ text annotation cues.
6. Where is the ball hit? From the specific camera
which has taken the shot.

We have extracted the following annotation cues:

1. Get a particular category of shot→ From shot classification results.
2. Get the highlights in video? → From algorithm wipes and motion templates alg.
3. Get the number of balls bowled → From the number of Bowl + Bat shots obtained from shot classification algorithm.

The performance of these annotation cues in video retrieval is same as the performance criteria of the mechanism which generates these cues. For example, the precision with which query to number of balls is answered is the precision with which Bowl+Bat shots are detected.

5.6 Applications of Automatic Annotation

As explained in chapter 2, manual annotation is tedious, time consuming, subjective, and a costly process. Hence there is a need to automatically annotate the sports videos. In the field of supporting technologies for the editorial process for both the old and new media, automatic annotation of video material opens the way to the economic exploitation of valuable assets. In particular, in the specific context of cricket videos, for the annotation cues which we have extracted as part of our work, there are some major applications which are briefed below:

- *Posterity logging*: This is known as one key method of improving production quality by bringing added depth and historical context to recent events. Posterity logging is typically performed by librarians to make a detailed annotation of the video tapes, according to some standard format. An example of posterity logging is the reuse of the shots that shows the best actions of a famous player; they can be reused later to provide a historical context.
- *Production logging*: where broadcasters use footage recorded few hours before, which may be even provided by a different broadcaster, and thus is not indexed, to annotate it in order to edit and produce a sports news program. Production logging is typically carried out live (or shortly after the event) by an assistant producer to select relevant shots to be edited into a magazine or news program that reports on sports highlights of the day. An example of production logging is the reuse of highlights, such as boundary, wicket, sixer etc to produce programs that contain the best sporting actions of the day.
- *Web based content retrieval*: Suppose the highlight actions or best actions of sports person are detected and allowed for browser based retrieval, then this saves lot of Bandwidth, due to not downloading the unnecessary video segments.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

The present work addressed the problem of automatic annotation of cricket videos keeping in view the applications it is intended for in both the broadcasting industry and content based video retrieval applications. Towards achieving this goal, we present here the conclusions of our research work.

- The first and foremost step in any video database management application, i.e., shot segmentation of video is implemented by a new algorithm to detect both abrupt and gradual transitions that are present in cricket videos. The algorithm has been implemented in a computationally efficient manner without compromising on overall system performance (of around 90%) on shot detection.
- We have implemented the task of automatic shot classification by considering two main low level image features i.e., color histograms and energy of difference images, where the second feature is used in a way to represent the motion in a camera shot. Our shot classification results by various combinations of above two feature templates have shown that Block based color templates are the best alternative. This yielded an overall system performance of around 50% in shot classification stage.
- We used the energies of frame differences to effectively generate a metric called shot energy, which is used along with the shot transitions wipes, to automatically detect the presence of slow motion shots, and thus facilitating the application of Highlights generation; one of the popular applications of semantic annotation of sports videos. The results of slow motion detection have an overall system performance of around 66%, which is considered as a reasonably decent figure.
- Also we generated some annotation cues like type of shot, number of balls being bowled, location of highlights by making use of results from shot classification.

6.2 Future Work

There is lot of scope for future work in this area of research, be it annotation of sports videos or in general the video database management systems, as there are a number of open research problems. For example complete and robust shot segmentation is still an open research problem. Similarly face detection irrespective of pose, illumination criteria, and content based image retrieval, robust image segmentation etc are some of the many other open research problems in this area.

As far as the present work is concerned, the following things can be tried to improve the performance of annotation.

- The algorithm on dissolves has to be improved by making use of cumulative differences of images, in a sensible manner so as not to increase computational complexity.
- Our shot classification step used only two features, color and motion, but many other features like shape, texture, optical flows, spatial moments etc can be made use of to increase the overall system performance.
- The use of statistical models cannot be ignored, and for this we need to first find the semantic structure in which shots occur, and hence find transition probabilities for each shot category. These probabilities can be used in a HMM framework by making use of Gaussian Mixture Models to model content variation across shots, both within a match and across matches.

BIBLIOGRAPHY

1. Jurgen Assfalg, Marco Bertini, Carlo Colombo, Alberto Del Bimbo, Semantic Annotation of Sports Videos, *Journal of IEEE Multimedia*, Vol 9, No. 2, pp. 52-60, 2002.
2. M. H. Kolekar, S. Sengupta, Hidden Markov Model Based Structuring of Cricket Video Sequences Using Motion and Color Features, *Proceedings of Fourth Indian Conference on Computer Vision, Graphics, and Image Processing(ICVGIP-2004)*, pp632-637, Kolkata, 2004.
3. R. Hjelsvold, Video STAR – A database for video information sharing, Dr. Ing.thesis, Norwegian Institute of Technology, November 1995.
4. K. Brown, A rich diet: Data-rich multimedia has a lot in store for archiving and storage companies, *Broadband Week*, March 5, 2001.
5. R. Bryll, A practical video database system, Master Thesis, University of Illinois at Chicago, 1998.
6. A. K. Elmagarmid, H. Jiang, A.A. Helal, A. Joshi, and M. Ahmed, *Video Database Systems*, Kluwer, Norwell, MA, 2002.
7. A. Dailianas, R. B. Allen, and P. England, Comparison of Automatic Video Segmentation Algorithms, in *Proceedings of SPIE Photonics West*, 1995. SPIE, Vol. 2615, pp.2-16, Philadelphia, 1995.
8. R. Dugad, K. Ratakonda, and N. Ahuja, Robust Video Shot Change Detection, *IEEE Second Workshop on Multimedia Signal Processing*, pp. 376-381, Redondo Beach, California, 1998.
9. R. M. Ford, C. Robson, D. Tample, and M. Gerlach, Metrics for Scene Change Detection in Digital Video Sequences, in *IEEE International Conference on Multimedia Computing and Systems (ICMCS'97)*, pp.610-611, Ottawa, Canada, 1997.
10. U. Garge, R. Kasturi, and S. H. Strayer, Performance Characterization of Video Shot Change Detection Methods, *IEEE Transactions of Circuits and Systems for Video Technology*, Vol. 10, No. 1, pp. 1-13, 2000.
11. D. Le Gall, A Video Compression Standard for Multimedia Applications, *Communications of the ACM*, Vol. 34, No. 4, pp.46-58, 1991.

12. C. W. Ngo, T. C. Pong, and R.T. Chin, Camera Breaks Detection by Partitioning of 2D Spatio-temporal Images in MPEG Domain, in *IEEE International Conference on Multimedia Systems (ICMS'99)*, vol. 1, pp.750-755, Italy, 1999.
13. N. V. Pathel and I. K. Sethi, Video Shot Detection and characterization for Video Databases, *Pattern Recognition, Special Issue on Multimedia*, Vol.30, pp. 583-592, 1997.
14. K. Sethi, and N.V. Patel, A Statistical Approach to Scene Change Detection, in *IS&T SPIE Proceedings: Storage and Retrieval for Image and Video Databases III*, Vol. 2420, pp. 329-339, San Jose, 1995.
15. L. Yeo and B. Liu, Rapid Scene Analysis on Compressed Video, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 5, No. 6, pp. 533-544, 1995.
16. H. J. Zhang, A. Kankanhalli, and S.W. Smoliar, Automatic Partitioning of Full-motion Video, *Multimed, ia Systems*, Vol. 1, No. 1, pp. 10-28, 1993.
17. C. W. Ngo, T. C. Pong, and R.T. Chin, Video Partitioning by Temporal Slice Coherency in *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 8, pp. 941-953, 2001.
- 18 J. M. Corridoni and A. Del Bimbo, Structured Representation and Automatic Indexing of Movie Information Content, *Pattern Recognition*, vol. 31, no. 12, pp. 2027-2045, 1998.
- 19 R. Zabih, J. Miller, and K. Mai, A Feature-Based Algorithm for Detecting and Classifying Scene Breaks, in *Proceedings of ACM Multimedia '95*, pp. 189-200, San Francisco, 1995.
20. A. Kokaram , P. Delacourt, A New Global Estimation Algorithm and its application to retrieval in sports events, *IEEE Fourth Workshop in Multimedia Signal Processing*, pp 251-256, Cannes, 2001.
21. C. Colombo, A. Del Bimbo, and P. Pala, " Semantics in Visual Information Retrieval", *IEEE Multimedia*, 6(3), 38-53, 1999.
22. Handbook of Video Databases: Design and Applications, Edited by Borko Furht, and Oge Marques, CRC Press, London, 2003.
23. J. Assfalg, M. Bertini, C. Colombo, and A. Del Bimbo, *Integrating Domain Knowledge and Visual Evidence to Support Highlight Detection in Sports Videos*, pp 97-121, Handbook of Video Databases: Design and Applications, edited by Borko Furht and Oge Marques. CRC press, London, 2003.

24. Hong Lu and Yap-Peng Tan, Content-based Sports Video based Analysis and Modeling, *Seventh International Conference on Control, Automation, Robotics, and Vision (ICARCV'02)*, December 2002, Singapore.
25. M.G. Kelly, K. M. Curtis, and M.P. Craven Fuzzy Recognition of Cricket Batting strokes based on Sequences of Body and Bat Postures, pp. 140-147, *Proceedings of IEEE SoutheastCon 2003*.
26. Peng Wang, Rui Cai, and Shi-Qiang Yang, Contextual Browsing for Highlights in Sports Videos, pp. 1951-1954, *IEEE International Conference on Multimedia and Expo (ICME)*, 2004.
27. Lei Wang, Xu Liu, Steve Lin, Guanyou Xu and Heung-Yeung Shum, Generic Slow Motion Replay Detection in Sports Videos, *IEEE International Conference on Image Processing (ICIP)*, pp 1585-88, 2004.